Fig. 1

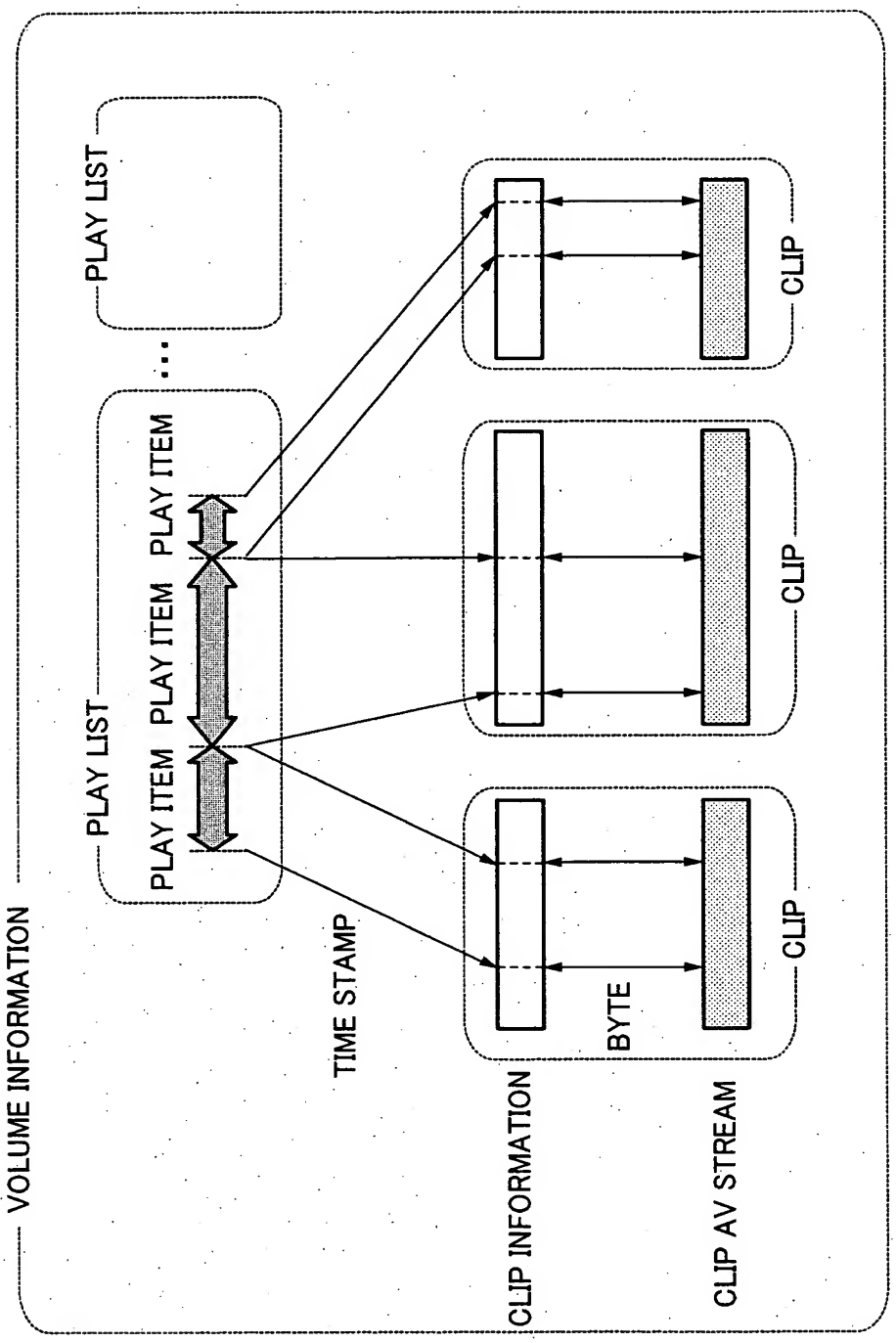

VOLUME INFORMATION
PLAY LIST
PLAY ITEM
PLAY ITEM
PLAY ITEM
PLAY LIST
TIME STAMP
CLIP INFORMATION
BYTE
CLIP AV STREAM
CLIP
CLIP
CLIP

Fig. 2

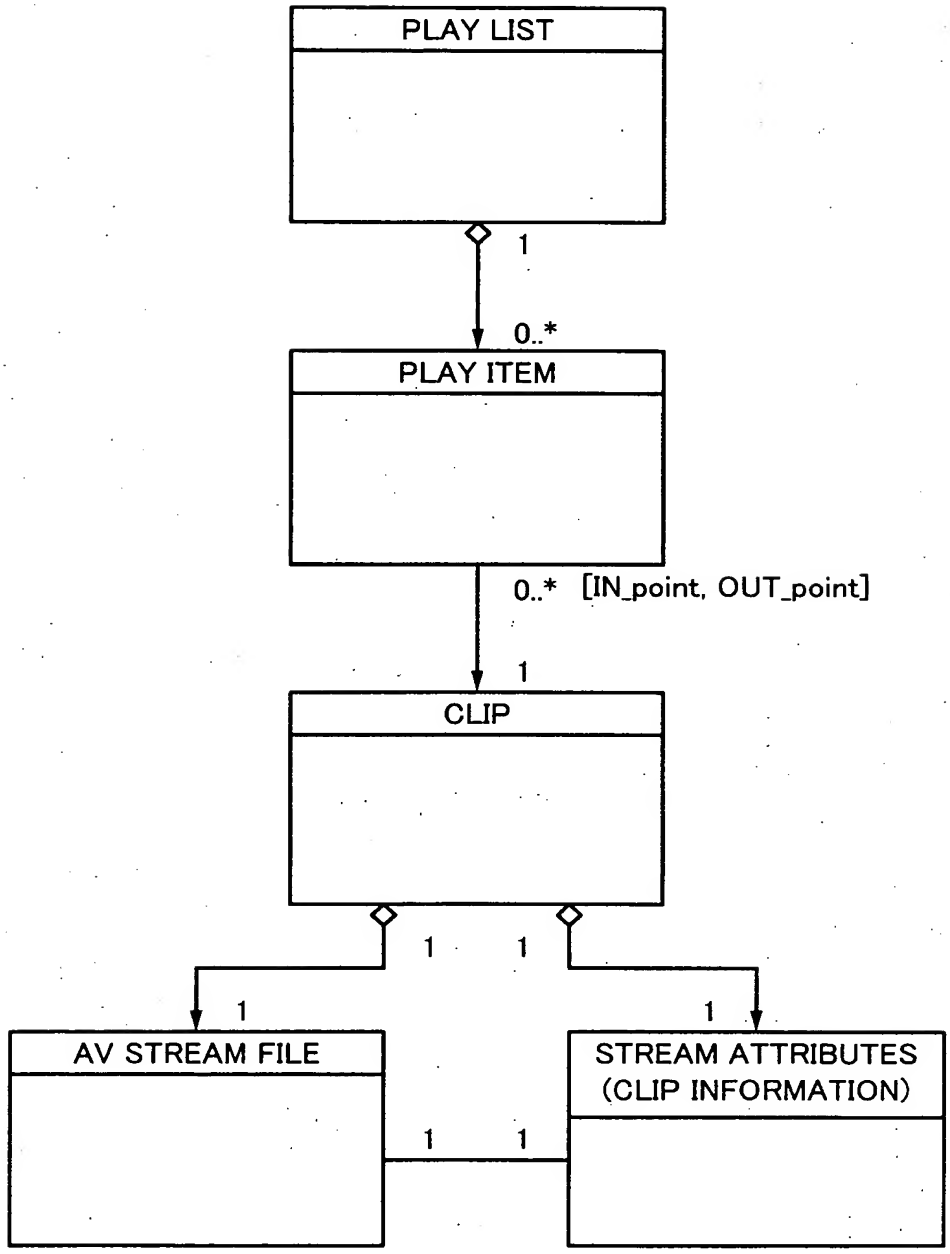

PLAY LIST
1
0..*
PLAY ITEM
0..* [IN_point, OUT_point]
1
CLIP
1
1
1
1
AV STREAM FILE
STREAM ATTRIBUTES
(CLIP INFORMATION)
1
1

Fig. 3

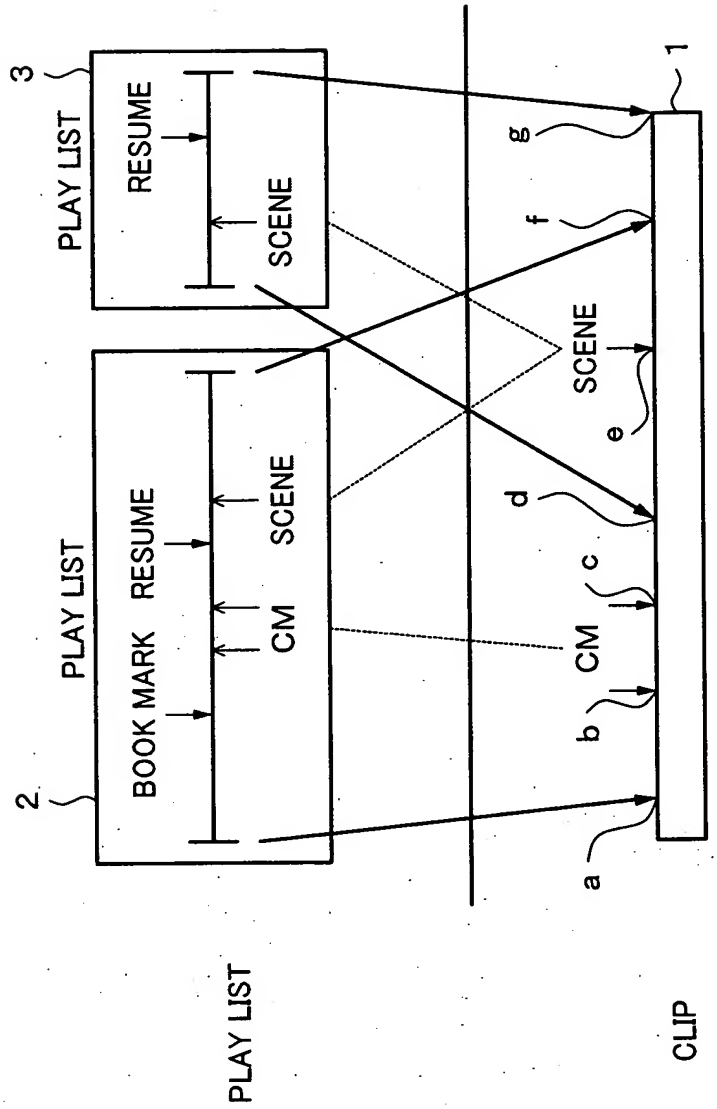

PLAY LIST
BOOK MARK
RESUME
CM
SCENE
2
PLAY LIST
RESUME
SCENE
3
a
b
CM
c
d
SCENE
e
f
g
1
PLAY LIST
CLIP

Fig. 4

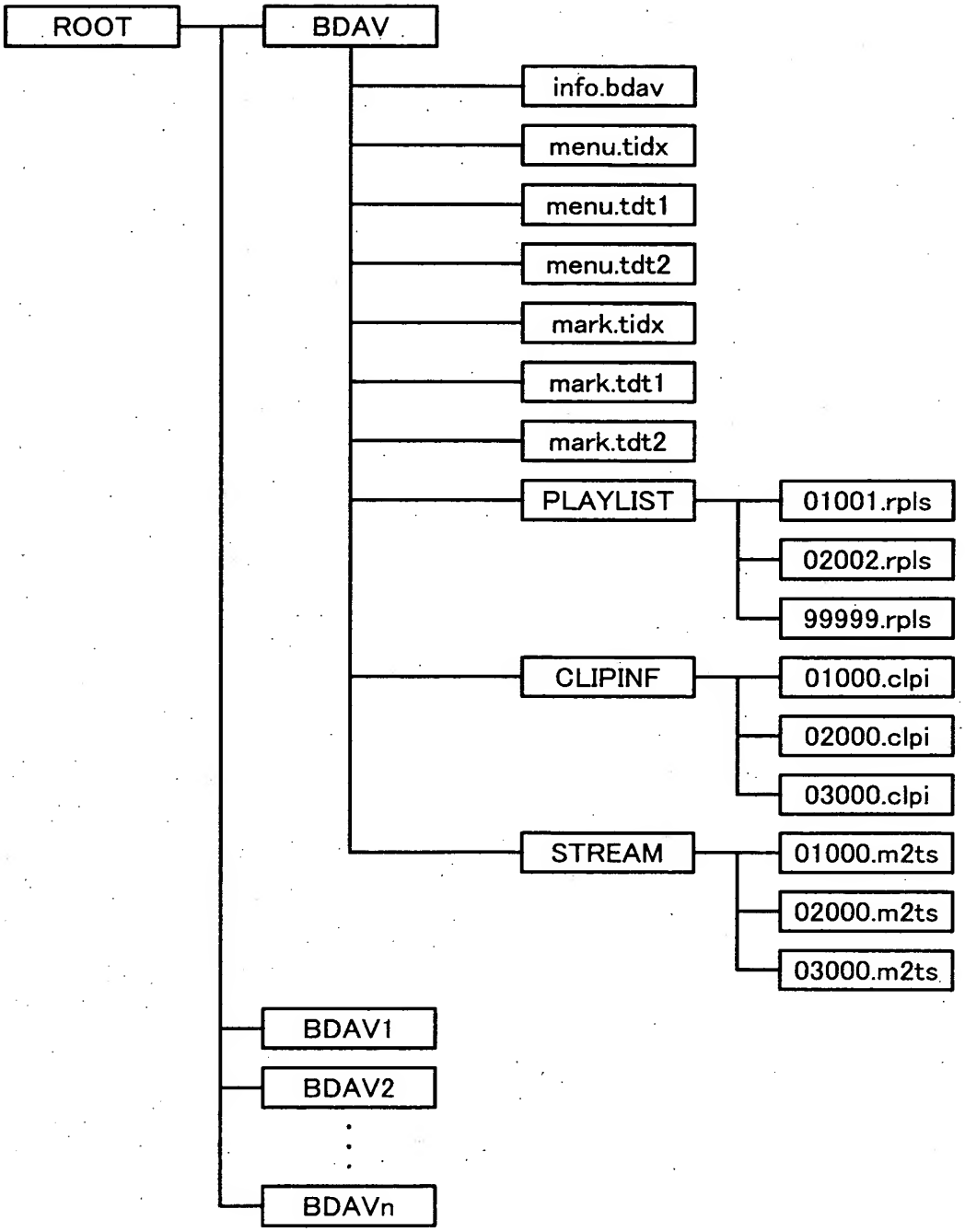

ROOT
BDAV
info.bdav
menu.tidx
menu.tdt1
menu.tdt2
mark.tidx
mark.tdt1
mark.tdt2
PLAYLIST
01001.rpls
02002.rpls
99999.rpls
CLIPINF
01000.clpi
02000.clpi
03000.clpi
STREAM
01000.m2ts
02000.m2ts
03000.m2ts
BDAV1
BDAV2
BDAVn

Fig. 5

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
info.bdav{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
TableOfPlayLists_start_address	32	unimsbf
MakersPrivateData_start_address	32	unimsbf
reserved_for_future_use	192	bslbf
UIAppInfoBDAV()		
for(i=0;i<N1;+i++){		
padding_word	16	bslbf
}		
TableOfPlayLists()		
for(i=0;i<N2;+i++){		
padding_word	16	bslbf
}		
MakersPrivateData()		
for(i=0;i<N3;+i++){		
padding_word	16	bslbf
}		
}		

Fig. 6

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
UIAppInfoBDAV(){		
length	32	unimsbf
reserved_for_future_use	16	bslbf
BDAV_character_set	8	bslbf
reserved_for_word_align	6	bslbf
BDAV_protect_flag	1	bslbf
resume_valid_flag	1	bslbf
PIN	8*4	bslbf
resume_PlayList_file_name	8*10	bslbf
ref_to_menu_thumbnail_index	16	unimsbf
BDAV_name_length	8	unimsbf
BDAV_name	8*255	bslbf
}		

Fig. 7

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
TableOfPlayLists(){		
length	32	unimsbf
number_of_PlayLists	16	unimsbf
for(i=0;i<*number_of_PlayLists*;i++){		
PlayList_file_name	8*10	bslbf
}		
}		

Fig. 8

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
xxxxx.rpls/yyyyy.vpls{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
PlayList_start_address	32	unimsbf
PlayListMark_start_address	32	unimsbf
MakersPrivateData_start_address	32	unimsbf
reserved_for_future_use	160	bslbf
UIAppInfoPlayList()		
for(i=0;i<N1;i++){		
padding_word	16	bslbf
}		
PlayList()		
for(i=0;i<N2;i++){		
padding_word	16	bslbf
}		
PlayListMark()		
for(i=0;i<N3;i++){		
padding_word	16	bslbf
}		
MakersPrivateData()		
for(i=0;i<N4;i++){		
padding_word	16	bslbf
}		
}		

Fig. 9

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
UIAppInfoPlayList(){		
length	32	unimsbf
reserved_for_future_use	16	bslbf
PlayList_character_set	8	unimsbf
reserved_for_word_align	4	bslbf
playback_protect_flag	1	bslbf
write_protect_flag	1	bslbf
is_played_flag	1	bslbf
is_edited_flag	1	bslbf
time_zone	8	bslbf
reserved_for_word_align	8	bslbf
record_time_and_date	4*14	bslbf
PlayList_duration	4*6	bslbf
maker_ID	16	unimsbf
maker_model_code	16	unimsbf
channel_number	16	unimsbf
reserved_for_word_align	8	bslbf
channel_name_length	8	unimsbf
channel_name	8*20	bslbf
PlayList_name_length	8	unimsbf
PlayList_name	8*255	bslbf
PlayList_detail_length	16	unimsbf
PlayList_detail	8*1200	bslbf
}		

Fig. 10

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
PlayList(){		
length	32	unimsbf
reserved_for_word_align	12	bslbf
PL_CPI_type	4	bslbf
number_of_PlayItems	16	unimsbf
if(<*Virtual-PlayList*>&&*PL_CPI_type==1*){		
number_of_SubPlayItems	16	unimsbf
}else{		
reserved_for_word_align	16	bslbf
}		
for(*PlayItem_id=0;PlayItem_id*<*number_of_PlayItems;PlayItem_id*++){		
PlayItem()		
}		
if(<*Virtual-PlayList*>&&*CPI_type==1*){		
for(i=0;i<*number_of_SubPlayItems*;i++){		
SubPlayItem()		
}		
}		
}		

Fig. 11

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
PlayItem(){		
length	16	unimsbf
Clip_Infomation_file_name	8*5	bslbf
Clip_codec_identifier	8*4	bslbf
reserved_for_future_use	6	bslbf
connection_condition	2	bslbf
if(*CPI_type*==1){		
ref_to_STC_id	8	unimsbf
}else{		
reserved_for_word_align	8	bslbf
}		
IN_time	32	unimsbf
OUT_time	32	unimsbf
if(<Virtual-PlayList>&&connection_condition==3){		
BridgeSequenceInfo()		
}		
}		

Fig. 12

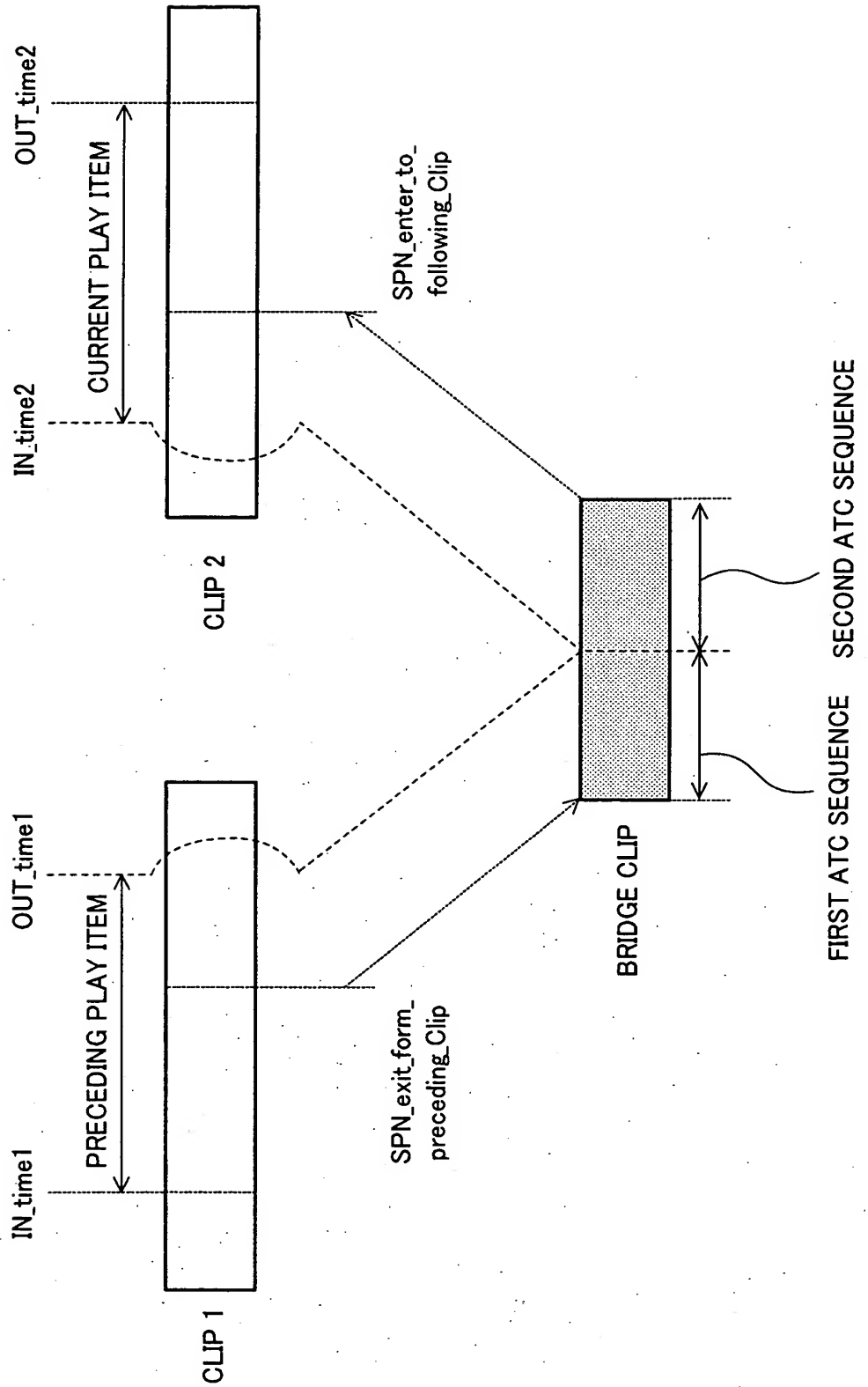

OUT_time2
CURRENT PLAY ITEM
IN_time2
CLIP 2
SPN_enter_to_following_Clip
OUT_time1
PRECEDING PLAY ITEM
IN_time1
CLIP 1
SPN_exit_form_preceding_Clip
BRIDGE CLIP
FIRST ATC SEQUENCE
SECOND ATC SEQUENCE

Fig. 13

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
PlayListMark(){		
length	32	unimsbf
number_of_PlayList_marks	16	unimsbf
for(i=0;i<number_of_PlayList_marks;i++){		
mark_invalid_flag	1	unimsbf
mark_type	7	unimsbf
mark_name_length	8	unimsbf
maker_ID	16	unimsbf
ref_to_PlayItem_id	16	unimsbf
mark_time_stamp	32	unimsbf
entry_ES_PID	16	unimsbf
if(mark_type==0x01||mark_type==0x02){		
ref_to_menu_thumbnail_index	16	unimsbf
}else{		
ref_to_menu_thumbnail_index	16	unimsbf
}		
duration	32	unimsbf
makers_infomation	32	bslbf
mark_name	8*24	bslbf
}		
}		

Fig. 14

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
zzzzz.clpi{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
SequenceInfo_start_address	32	unimsbf
ProgramInfo_start_address	32	unimsbf
CPI_start_address	32	unimsbf
ClipMark_start_address	32	unimsbf
MakersPrivateData_start_address	32	unimsbf
reserved_for_future_use	96	bslbf
ClipInfo()		
for(i=0;<N1;i++){		
padding_word	16	bslbf
}		
SequenceInfo()		
for(i=0;<N2;i++){		
padding_word	16	bslbf
}		
ProgramInfo()		
for(i=0;<N3;i++){		
padding_word	16	bslbf
}		
CPI()		
for(i=0;<N4;i++){		
padding_word	16	bslbf
}		
ClipMark()		
for(i=0;<N5;i++){		
padding_word	16	bslbf
}		
MakersPrivateData()		
for(i=0;<N6;i++){		
padding_word	16	bslbf
}		
}		

Fig. 15

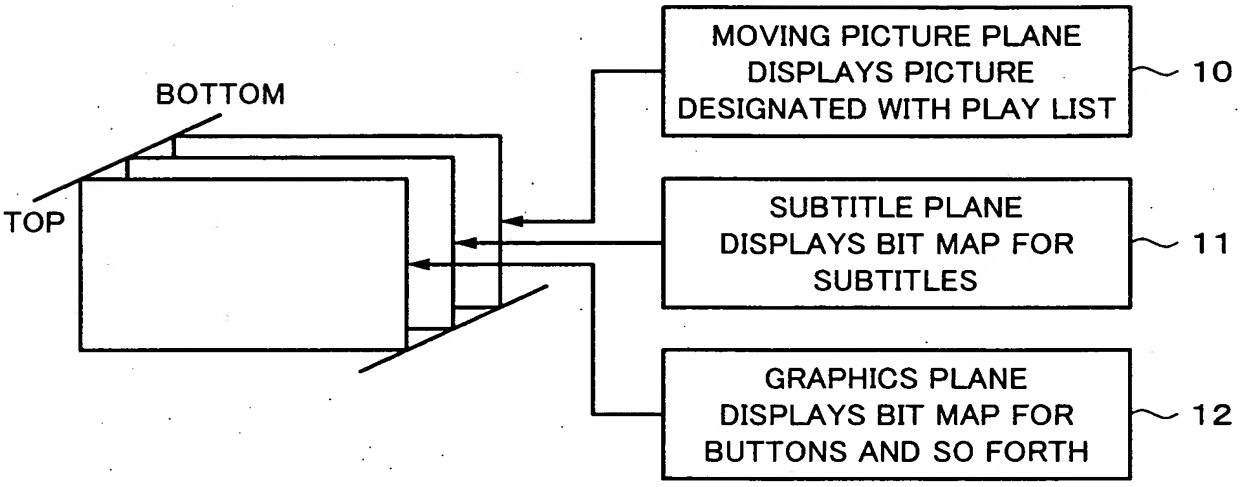


Fig. 16

ITEM	DESCRIPTION
MOVING PICTURE PLANE	1920x1080x16BITS, YCbCr(4:2:2),EIGHT BITS EACH
SUBTITLE PLANE	1920x1080x8BITS, 8-BIT COLOR MAP ADDRESSES (PALETTE) + ALPHA-BLENDING IN 256 LEVELS
GRAPHICS PLANE	1920x1080x8BITS, 8-BIT COLOR MAP ADDRESSES (PALETTE) + ALPHA BLENDING IN 256 LEVELS

Fig. 17

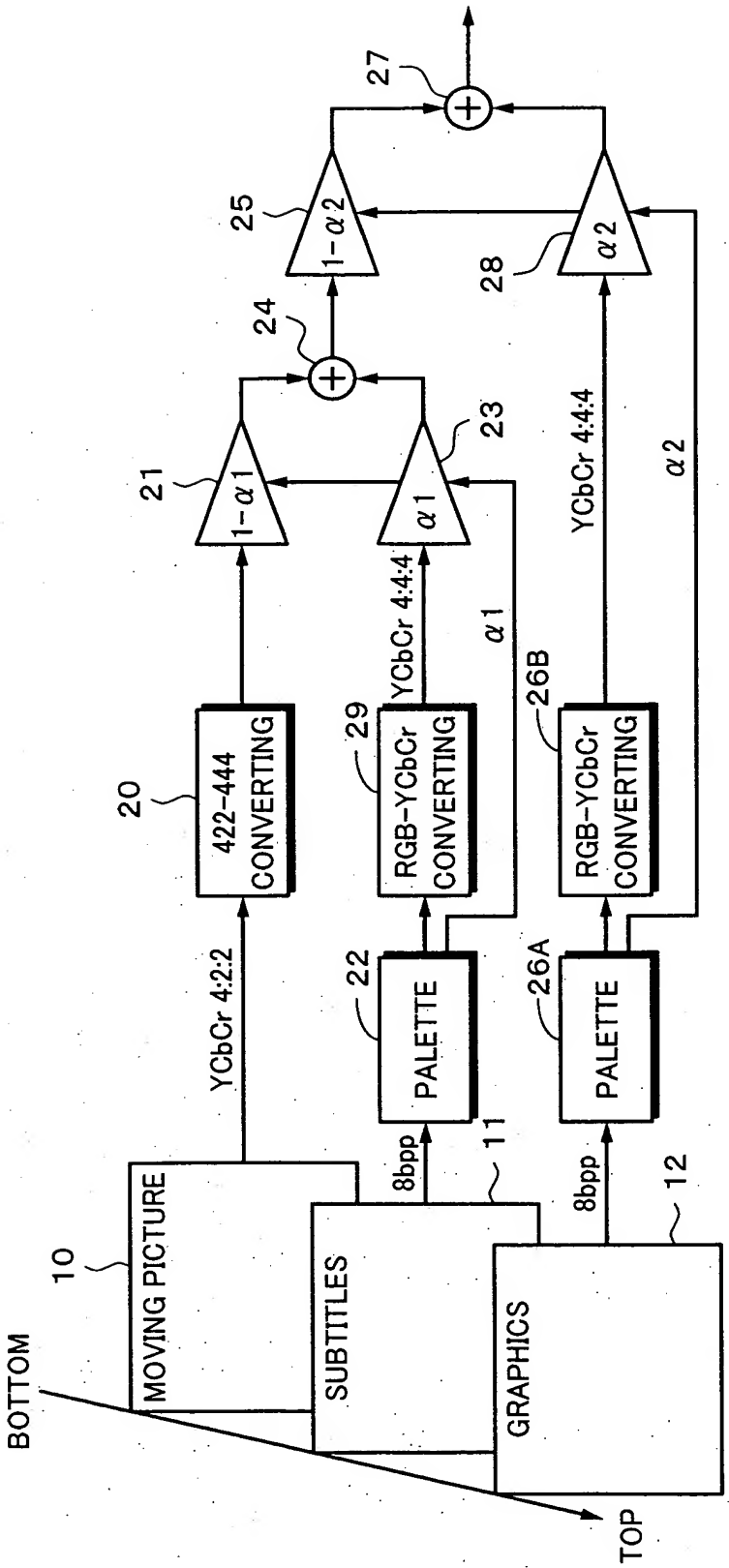

BOTTOM
TOP
10
MOVING PICTURE
SUBTITLES
GRAPHICS
11
12
8bpp
8bpp
YCbCr 4:2:2
22
PALETTE
26A
PALETTE
20
422-444 CONVERTING
29
RGB-YCbCr CONVERTING
26B
RGB-YCbCr CONVERTING
YCbCr 4:4:4
YCbCr 4:4:4
α1
α2
21
1-α1
23
α1
24
25
1-α2
28
α2
27

Fig. 18

INPUT	INPUT ADDRESS, 8 BITS
OUTPUT	OUTPUT DATA, 8 BITS×4, (R, G, B, α) OUTPUT

Fig. 19

	VALUES OF THREE PRIMARY COLORS			TRANSPARENCY
COLOR INDEX VALUE	R	G	B	α
0x00	0	0	0	0
0x01	10	100	30	0.5
⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮
0xFF	200	255	100	0.8

Fig. 20

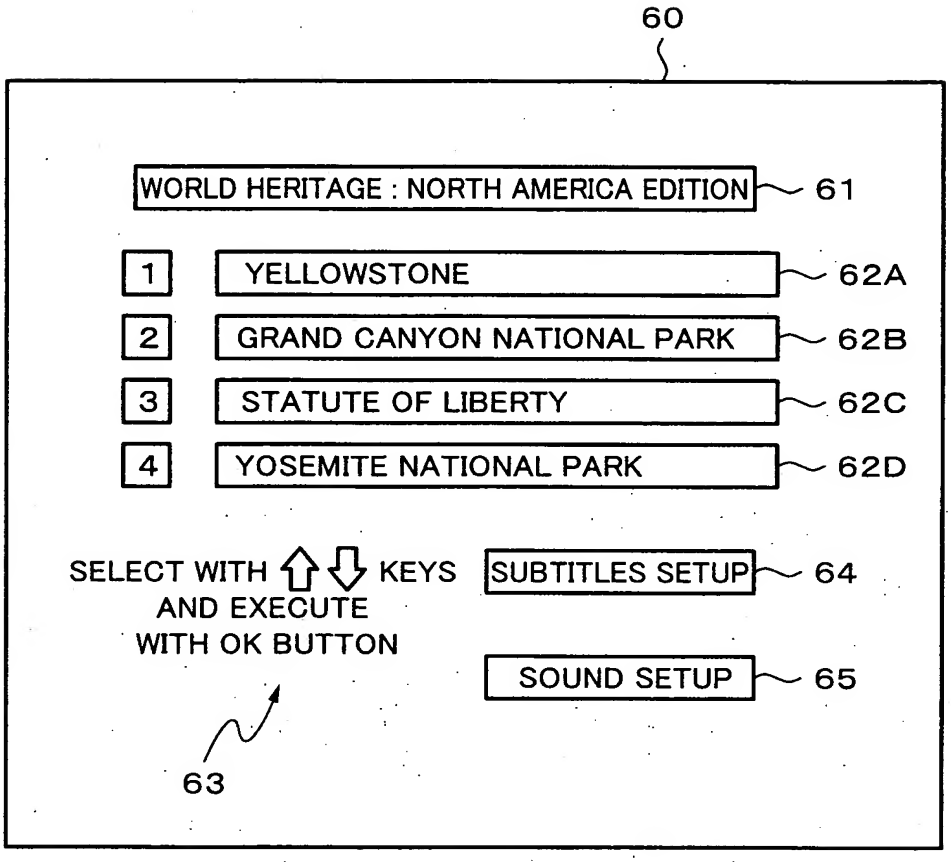

60
WORLD HERITAGE : NORTH AMERICA EDITION
61
1
YELLOWSTONE
62A
2
GRAND CANYON NATIONAL PARK
62B
3
STATUTE OF LIBERTY
62C
4
YOSEMITE NATIONAL PARK
62D
SELECT WITH ⇧⇩ KEYS
AND EXECUTE
WITH OK BUTTON
63
SUBTITLES SETUP
64
SOUND SETUP
65

Fig. 21

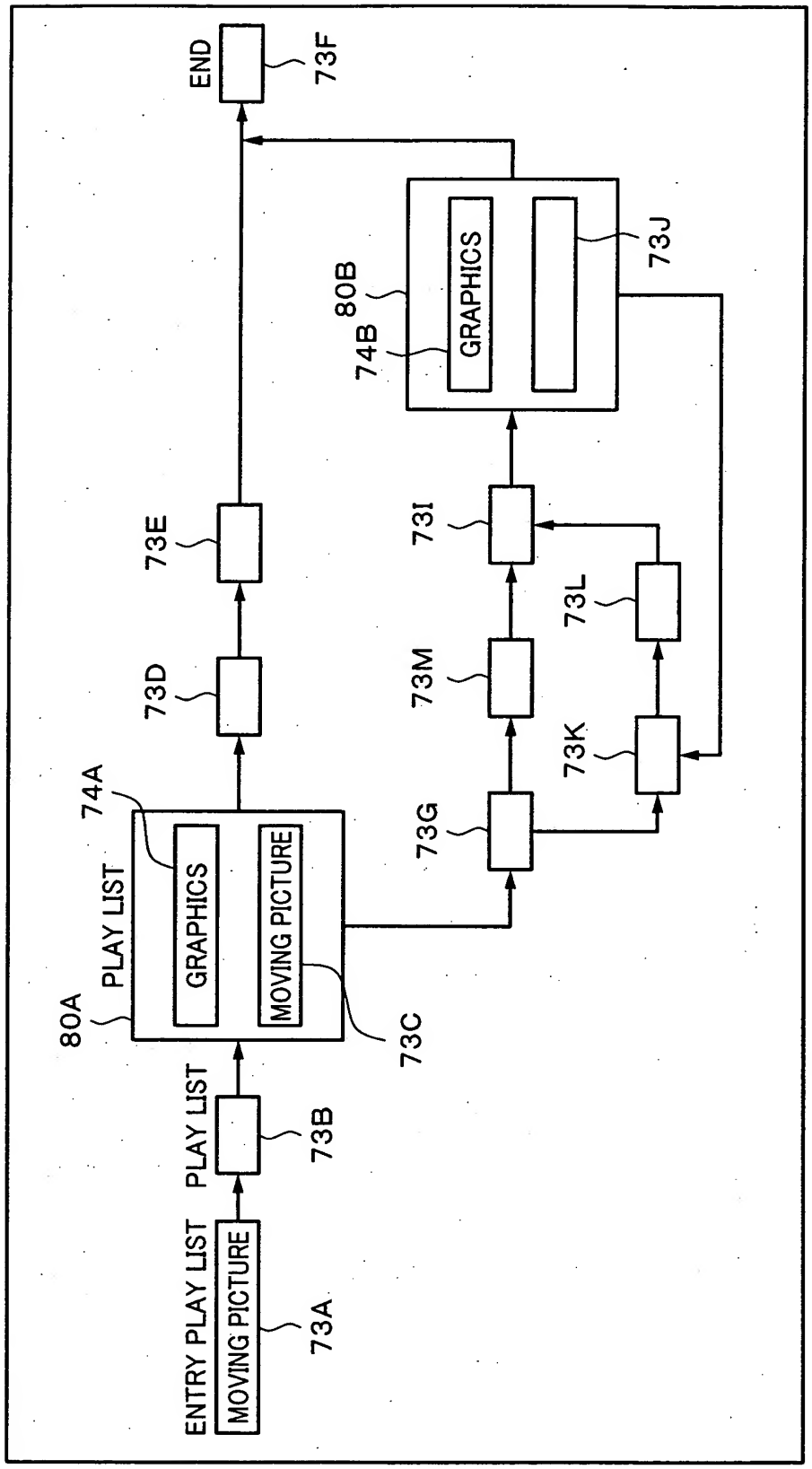

ENTRY PLAY LIST
MOVING PICTURE
73A
PLAY LIST
73B
80A
PLAY LIST
GRAPHICS
MOVING PICTURE
74A
73C
73D
73E
END
73F
73G
73M
73K
73L
73I
80B
74B
GRAPHICS
73J

Fig. 22

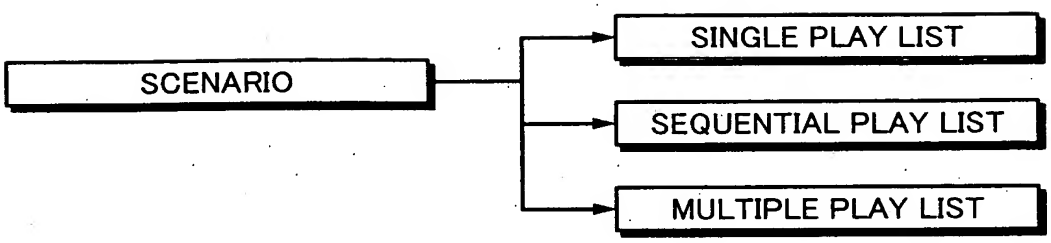


Fig. 23A

Fig. 23B

Fig. 23C

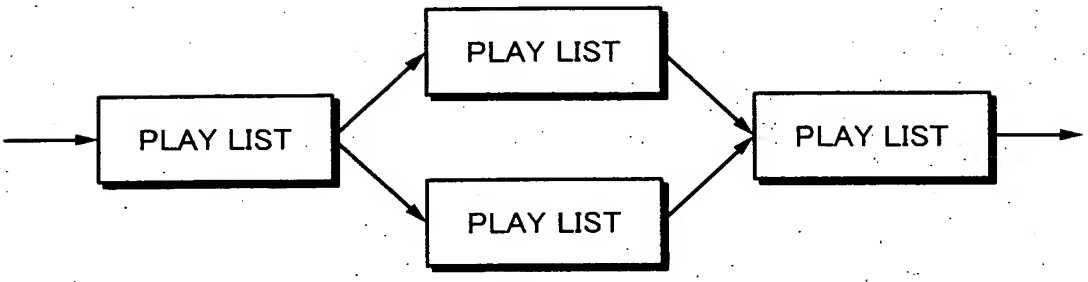


Fig. 24

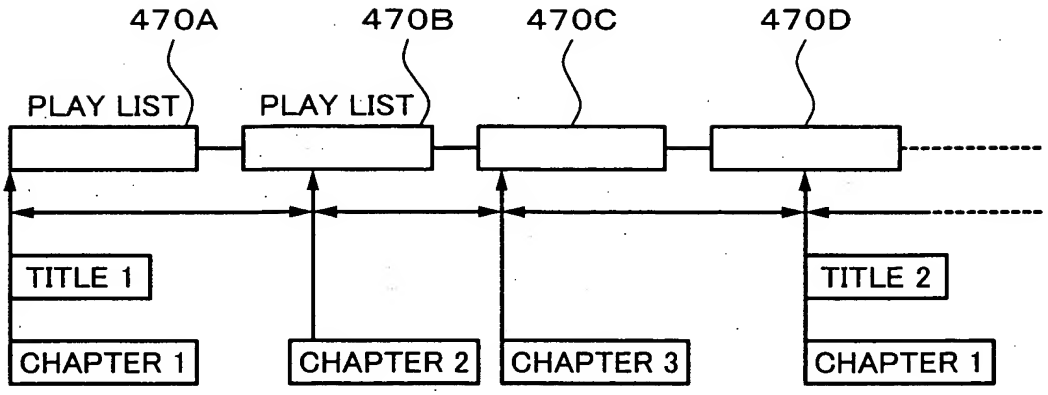
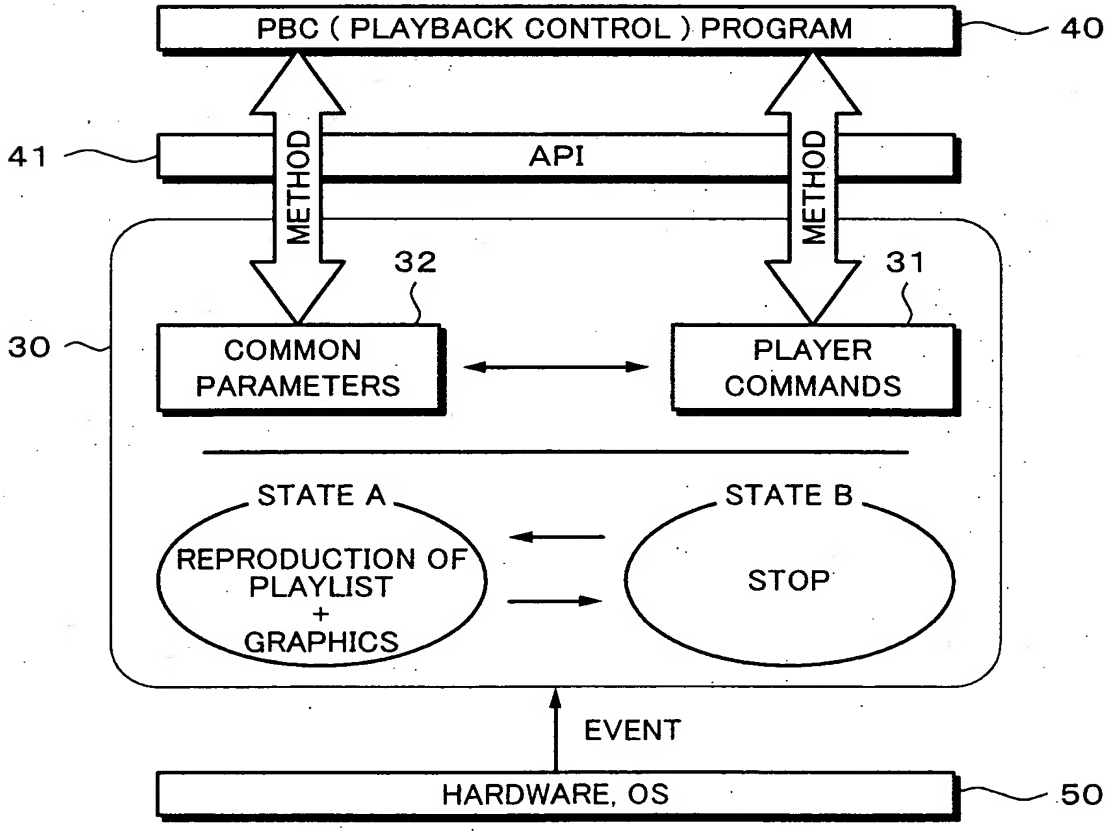

470A
470B
470C
470D
PLAY LIST
PLAY LIST
TITLE 1
TITLE 2
CHAPTER 1
CHAPTER 2
CHAPTER 3
CHAPTER 1

Fig. 25


PBC (PLAYBACK CONTROL) PROGRAM
40
41
API
METHOD
METHOD
32
31
30
COMMON PARAMETERS
PLAYER COMMANDS
STATE A
STATE B
REPRODUCTION OF PLAYLIST + GRAPHICS
STOP
EVENT
HARDWARE, OS
50

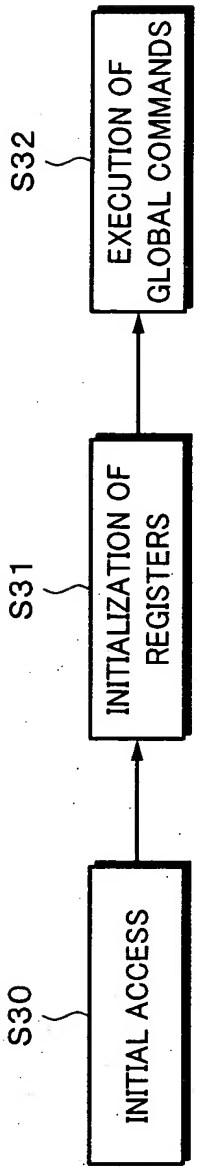


Fig. 26A

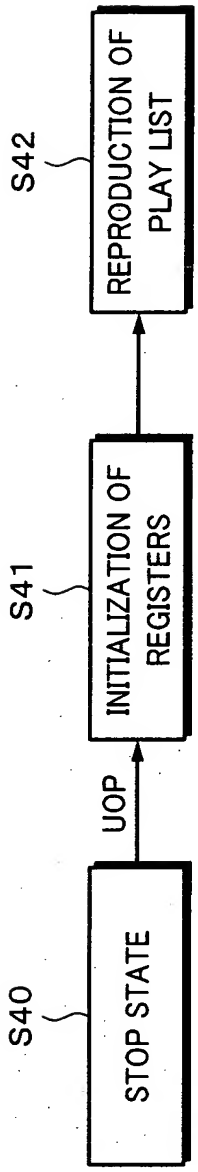


Fig. 26B

Fig. 27A

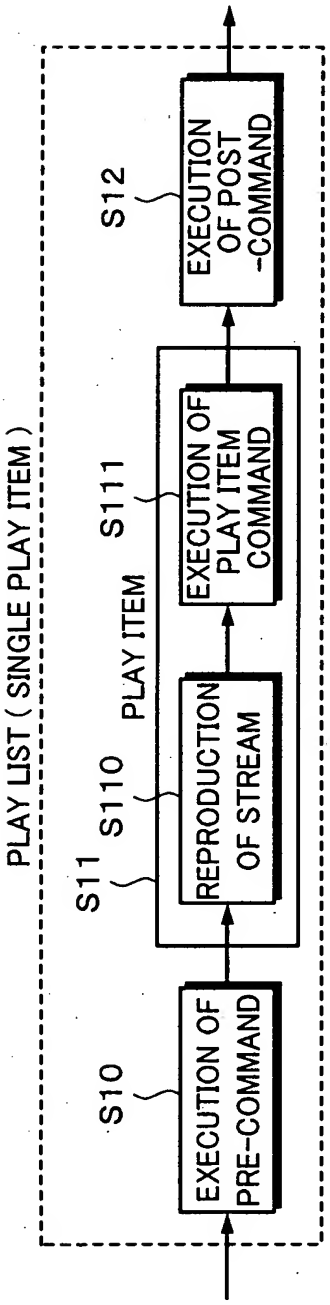


Fig. 27B

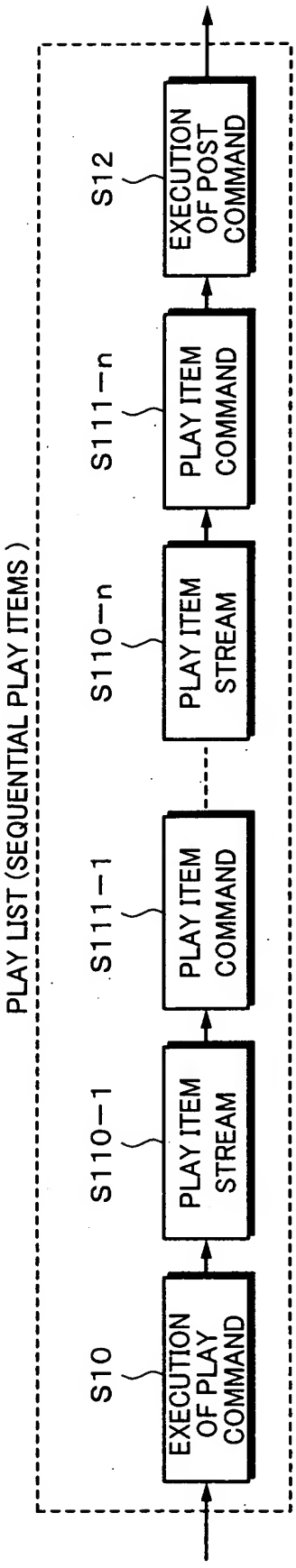


Fig. 28A

Fig. 28

Fig. 28A
Fig. 28B
Fig. 28C
Fig. 28D
Fig. 28E
Fig. 28F
Fig. 28G
Fig. 28H

METHOD	REMARKS
COMMANDS FOR DESIGNATING REPRODUCTION START POSITION	
LinkPlayList(playListNumber)	CAUSES REPRODUCTION OF PlayList DESIGNATED BY PlayListNumber TO BE STARTED.
LinkPlayItem(playListNumber,playItemNumber)	CAUSES REPRODUCTION OF DESIGNATED PlayItem OF DESIGNATED PlayList TO BE STARTED. playItemNumber IS PlayItem_id STARTING FROM 0. WHEN PlayList IS REPRODUCED FROM BEGINNING, PlayItemNumber IS 0.

Fig. 28B

METHOD	REMARKS
COMMANDS FOR DESIGNATING REPRODUCTION START POSITION	
Link(position)(object) position=("prev"|"next"|"top"|"Parent"|"tail") object=(PlayList | PlayItem | Chapter)	CAUSES CURRENT POSITION TO BE MOVED IN SCENARIO. CAUSES CURRENT REPRODUCTION POSITION TO BE MOVED TO ADJACENT PalyList, PlayItem, OR Chapter.
Exit	CAUSES REPRODUCTION OF SCENARIO TO BE STOPPED. VALUE OF STANDARD REGISTER IS NOT HELD.
RSM	CAUSES REPRODUCTION TO BE RESUMED FROM LAST REPRODUCTION STOP POSITION. CAUSES STORED RESUME INFORMATION TO BE CALLED, IT TO BE SET TO REGISTER, AND REPRODUCTION OF SCENARIO TO BE STARTED.

Fig. 28C

COMMANDS FOR OBTAINING STATE OF PAYER		
	getMenuDescriptionLanguage()	CAUSES LANGUAGE OF MENU THAT IS DISPLAYED TO BE OBTAINED.
	getScenarioNumber()	CAUSES SCENARIO NUMBER THAT IS BEING REPRODUCED TO BE OBTAINED.
	getPlayListNumber()	CAUSES PLAY LIST NUMBER THAT IS BEING REPRODUCED TO BE OBTAINED.
	getChapterNumber()	CAUSES CHAPTER NUMBER THAT IS BEING REPRODUCED TO BE OBTAINED.
	getPlayerSupport()	CAUSES VERSION AND FUNCTION OF PLAYER TO BE OBTAINED.

Fig. 28D

COMMANDS FOR VIDEO STREAMS	
getVideoStreamAvailability()	CAUSES INFORMATION THAT DESCRIBES WHETHER OR NOT DESIGNATED VIDEO STREAM IS CONTAINED TO BE OBTAINED.
setVideoStreamNumber()	DESCRIBES VIDEO STREAM TO BE DECODED.
getVideoStreamNumber()	CAUSES VIDEO STREAM NUMBER THAT IS BEING SELECTED TO BE OBTAINED.
getVideoStreamAttribute()	CAUSES ATTRIBUTE OF VIDEO STREAM (ENCODING SYSTEM, RESOLUTION, ASPECT RATIO, DISPLAY MODE IN THE CASE OF ASPECT RATIO OF 4 : 3, CLOSED CAPTION)TO BE OBTAINED.
setAngleNumber()	DESCRIBES ANGLE NUMBER.
getAngleNumber()	CAUSES ANGLE NUMBER THAT IS BEING SELECTED TO BE OBTAINED.
getMaxVideoStreams()	CAUSES NUMBER OF VIDEO STREAMS THAT CAN BE SELECTED TO BE OBTAINED. DESCRIBES WHETHER OR NOT getVideoStreamAvailability() IS SUFFICIENT.

Fig. 28E

COMMANDS FOR AUDIO STREAMS	
getAudioStreamAvailability()	CAUSES INFORMATION THAT DESCRIBES WHETHER OR NOT DESIGNATED AUDIO STREAM IS CONTAINED TO BE OBTAINED.
getAudioStreamLanguage()	CAUSES INFORMATION ABOUT LANGUAGE OF DESIGNATED AUDIO STREAM TO BE OBTAINED.
getAudioStreamStatus()	DESCRIBES AUDIO STREAM TO BE REPRODUCED.
setAudioStreamStatus()	CAUSES AUDIO STREAM NUMBER THAT IS BEING REPRODUCED TO BE OBTAINED.
getAudioStreamAttribute()	CAUSES ATTRIBUTE OF AUDIO STREAM (ENCODING SYSTEM, NUMBER OF CHANNELS, Q, FS) TO BE OBTAINED.

Fig. 28F

COMMANDS FOR SUB PICTURE STREAMS	
getSPStreamAvailability()	CAUSES INFORMATION THAT DESCRIBES WHETHER OR NOT DESIGNATED SP STREAM IS CONTAINED TO BE OBTAINED.
getSPStreamLanguage()	CAUSES LANGUAGE OF DESIGNATED SP STREAM TO BE OBTAINED.
getSPDisplayStatus()	CAUSES DISPLAY STATE OF SP (WHETHER OR NOT SP IS DISPLAYED) TO BE OBTAINED.
setSPDisplayStatus()	DESCRIBES DISPLAY STATE OF SP (WHETHER OR NOT SP IS DISPLAYED).
getSpStreamAttribute()	CAUSES ATTRIBUTE OF SP (RESOLUTION, 4 : 3 OR WIDE) TO BE OBTAINED.

Fig. 28G

COMMANDS FOR REGISTER READ/WRITE	
clearReg()	CAUSES ALL REGISTERS TO BE INITIALIZED.
setReg()	CAUSES VALUE TO BE SET TO REGISTER.
getReg()	CAUSES VALUE TO BE READ FROM REGISTER.

Fig. 28H

COMMANDS FOR TIMERS	
sleep()	CAUSES PROCESS TO BE STOPPED FOR DESIGNATED MILLISECONDS.
setTimeout()	CAUSES FUNCTION AND PROCESS TO BE EXECUTED AFTER DESIGNATED MILLISECONDS HAVE ELAPSED.
setInterval()	CAUSES PROCESS TO BE EXECUTED AT INTERVALS OF DESIGNATED MILLISECONDS.
clearTimer()	CAUSES PROCESS OF TIMER THAT HAS DESIGNATED REGISTRATION TIMER ID TO BE STOPPED.
pauseTimer()	CAUSES TIMER THAT HAS DESIGNATED REGISTRATION TIMER ID TO BE TEMPORARILY STOPPED.
resumeTimer()	CAUSES TIMER THAT HAS DESIGNATED REGISTRATION TIMER ID TO BE RESUMED FROM PAUSE STATE.
OTHER COMMANDS	
playSoundEffect(sound_id)	CAUSES SELECTED EFFECT SOUND TO BE REPRODUCED. USED WITH BUTTON COMMAND.

Fig. 29

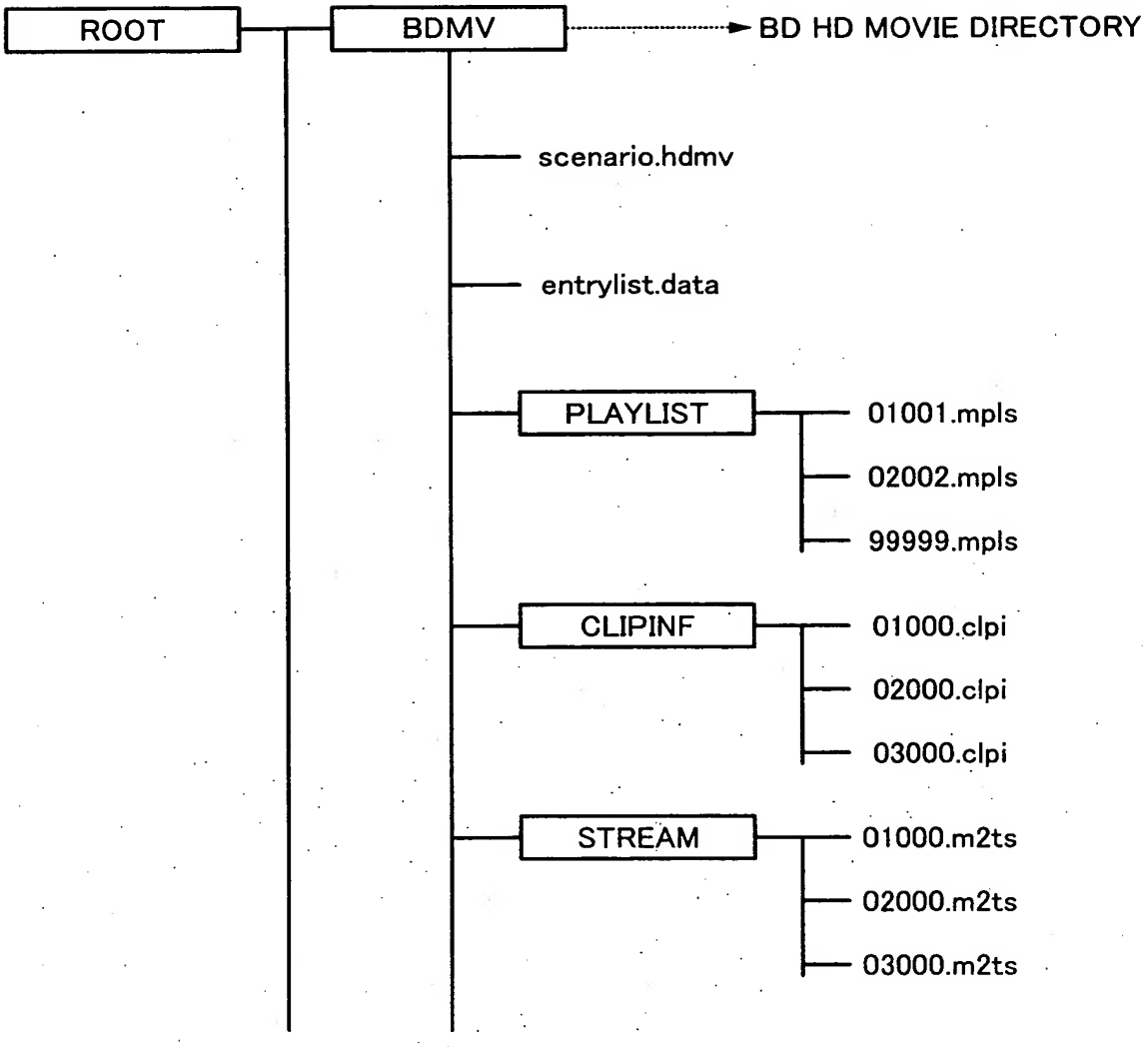

ROOT
BDMV
BD HD MOVIE DIRECTORY
scenario.hdmv
entrylist.data
PLAYLIST
01001.mpls
02002.mpls
99999.mpls
CLIPINF
01000.clpi
02000.clpi
03000.clpi
STREAM
01000.m2ts
02000.m2ts
03000.m2ts

Fig. 30

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
scenario.hdmv{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
scenario_start_address	32	
reserved_for_future_use	224	bslbf
Autoplay()		
for(i=0;i<N1;i++){		
padding_word	16	bslbf
}		
Scenario()		
for(i=0;i<N2;i++){		
padding_word	16	bslbf
}		
}		

Fig. 31

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
Autoplay(){		
length	32	uimsbf
reserved	16	
number_of_commands	16	
for(i=0;i<number_of_commands;i++){		
command(i)	32	uimsbf
}		
}		

Fig. 32

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
Scenario(){		
length	32	
flags	32	
number_of_PlayLists	16	
for(i=0;i<Number_of_PlayLists;i++){		
Pre_Command_start_id	32	
Post_Command_start_id	32	
number_of_Pre_Commands	32	
number_of_Post_Commands	32	
reserved	32	
number_of_PlayItems	32	
for(PlayItem_id=0;PlayItem_id<number_of_PlayItems;PlayItem_id++){		
PI_Command_start_id	32	
number_of_PI_Commands	32	
}		
reserved		
// Command table for fach PlayList		
number_of_PL_Commands	16	
for(j=0;j<number_of_PL_Commands;j++){		
PL_Command(j)	32	
}		
}		
}		

Fig. 33

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
entrylist.data{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
ScenarioEntry_start_address	32	uimsbf
reserved_for_future_use	224	bslbf
AppInfo()		
for(i=0;i<N1;i++){		
padding_word	16	bslbf
}		
ScenarioEntry()		
for(i=0;i<N2;i++){		
padding_word	16	bslbf
}		
}		

Fig. 34

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
AppInfo(){		
length	32	uimsbf
reserved_for_future_use	16	bslbf
HDMV_name_character_set	8	bslbf
reserved_for_word_align	7	bslbf
PIN_valid_flag	1	bslbf
PIN	8*4	bslbf
// UOP_mask_table() // For directory	64	
HDMV_name_length	8	uimsbf
HDMV_name	8*255	bslbf
}		

Fig. 35

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
ScenarioEntry(){		
length	32	unimsbf
name_character_set	8	bslbf
// Entry PL for the Top Menu		
Top Menu PL(){		
flags	32	bslbf
TopMenu_ref_to_PlayList_file_name	8*10	bslbf
TopMenu_ref_to_PlayItem_id	16	unimsbf
TopMenu_name_length	8	unimsbf
TopMenu_name	8*255	bslbf
}		
// Title Entries		
number_of_Titles	16	unimsbf
for(unit title_number=0;title_number<Number_of_Titles;title_number++){		
flags	32	bslbf
Title_ref_to_PlayList_file_name	8*10	bslbf
Title_ref_to_PlayItem_id	16	unimsbf
Title_name_length	8	unimsbf
Title_name	8*255	bslbf
}		
// Stream Setup Menu for each PL		
number_of_PlayLists	16	unimsbf
for(i=0;i<Number_of_PlayLists;i++){		
SSMenu_flags	32	bslbf
SSMenu_ref_to_PlayList_file_name	8*10	bslbf
SSMenu_ref_to_PlayItem_id	16	unimsbf
}		
}		

Fig. 36

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
xxxxx.mpls{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
PlayList_start_address	32	unimsbf
PlayListMark_start_address	32	unimsbf
reserved_for_future_use	192	bslbf
PLControlInfo()		
for(i=0;i<N1;i++){		
padding_word	16	bslbf
}		
PlayList()		
for(i=0;i<N2;i++){		
padding_word	16	bslbf
}		
PlayListMark()		
for(i=0;i<N3;i++){		
padding_word	16	bslbf
}		
}		

Fig. 37

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
PLControlInfo(){		
length	32	unimsbf
reserved_for_future_use	8	bslbf
PlayList_character_set	8	unimsbf
reserved_for_future_use	8	
PL_playback_type	8	
if(PL_playback_type==0x2 ||		
PL_playback_type==0x3){		
playback_count	16	
}else{		
reserved_for_word_align	16	
}		
PL_UOP_mask_table() // For PlayList	64	
reserved_for_word_align	8	
PL_random_access_mode	8	
reserved_for_word_align	8	bslbf
PlayList_duration	4*6	bslbf
PlayList_name_length	8	unimsbf
PlayList_name	8*255	bslbf
PlayList_detail_length	16	unimsbf
PlayList_detail	8*1200	bslbf
}		

Fig. 38

PL_playback_type	DESCRIPTION
0x0	RESERVED REGION
0x1	SEQUENTIALLY REPRODUCES PLAY ITEMS (NORMAL REPRODUCTION).
0x2	RANDOMLY REPRODUCES PLAY ITEMS.
0x3	SHUFFLE-REPRODUCES PLAY ITEMS.

Fig. 39

PL_random_access_mode	DESCRIPTION
0x0	PERMITS JUMP-REPRODUCTION AND VARIABLE SPEED REPRODUCTION.
0x1	PROHIBITS JUMP-REPRODUCTION AND VARIABLE SPEED REPRODUCTION.

Fig. 40

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
PlayList(){		
length	32	unimsbf
number_of_PlayItems	16	unimsbf
number_of_SubPlayItems	16	unimsbf
for(PlayItem_id=0;PlayItem_id<number_of_PlayItems;PlayItem_id++){		
PlayItem()		
}		
for(SubPlayItem_id=0;SubPlayItem_id<number_of_SubPlayItems;SubPlayItem_id++){		
SubPlayItem()		
}		
}		

Fig. 41

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
PlayItem(){		
length	16	uimsbf
reserved_for_word_align	8	bslbf
Clip_Information_file_name	8*5	bslbf
Clip_codec_identifier	8*4	bslbf
reserved_for_future_use	7	bslbf
is_multi_angle	1	bslbf
reserved_for_future_use	4	bslbf
connection_condition	4	uimsbf
ref_to_STC_id	8	uimsbf
IN_time	32	uimsbf
OUT_time	32	uimsbf
PI_UOP_mask_table()	64	bslbf
PID_filter()		
reserved_for_word_align	8	bslbf
PI_random_access_mode	8	uimsbf
reserved_for_word_align	8	bslbf
still_mode	8	uimsbf
if(still_mode==0x1){		
still_time	16	uimsbf
}else{		
reserved_for_word_align	16	bslbf
}		
// Angle		
if(is_multi_angle){		
number_of_angles	8	uimsbf
is_seamless_angle_change	8	uimsbf
for(angle_id=1;angle_id<number_of_angles;angle_id++){		
Clip_Information_file_name	8*5	bslbf
ref_to_STC_id	8	uimsbf
IN_time	32	uimsbf
OUT_time	32	uimsbf
}		
}		
}		

Fig. 42

PI_random_access_mode	DESCRIPTION
0x0	PERMITS JUMP REPRODUCTION AND VARIABLE SPEED REPRODUCTION.
0x1	PROHIBITS JUMP REPRODUCTION AND VARIABLE SPEED REPRODUCTION.

Fig. 43

still_mode	DESCRIPTION
0x0	NO STILL.
0x1	STILL FOR LIMITED TIME PERIOD. TIME PERIOD IS DESCRIBED IN still_time.
0x2	STILL FOR UNLIMITED TIME PERIOD. STILL IS CONTINUED UNTIL USER CANCELS IT.
0x3-0xf	RESERVED.

Fig. 44

is_seamless_angle_change	DESCRIPTION
0x0	NONSEAMLESSLY CHANGEABLE ANGLES
0x1	SEAMLESSLY CHANGEABLE ANGLES

Fig. 45

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
SubPlayItem(){		
length	16	unimsbf
Clip_Information_file_name	8*5	bslbf
Clip_codec_identifier	8*4	bslbf
reserved_for_future_use	7	bslbf
is_repeat_flag	1	bslbf
SubPlayItem_type	8	bslbf
ref_to_STC_id	8	unimsbf
SubPlayItem_IN_time	32	unimsbf
SubPlayItem_OUT_time	32	unimsbf
if(is_repeat_flag==0){		
sync_PlayItem_id	16	unimsbf
sync_start_PTS_of_PlayItem	32	unimsbf
}else{		
reserved_for_word_align	16	
reserved_for_word_align	32	
}		
}		

Fig. 46

is_repeat_flag	DESCRIPTION
0	PERFORMS REPRODUCTION IN SYNCHRONIZATION WITH MAIN PATH.
1	DOES NOT PERFORM REPRODUCTION IN SYNCHRONIZATION WITH MAIN PATH. REPEATS REPRODUTION.

Fig. 47

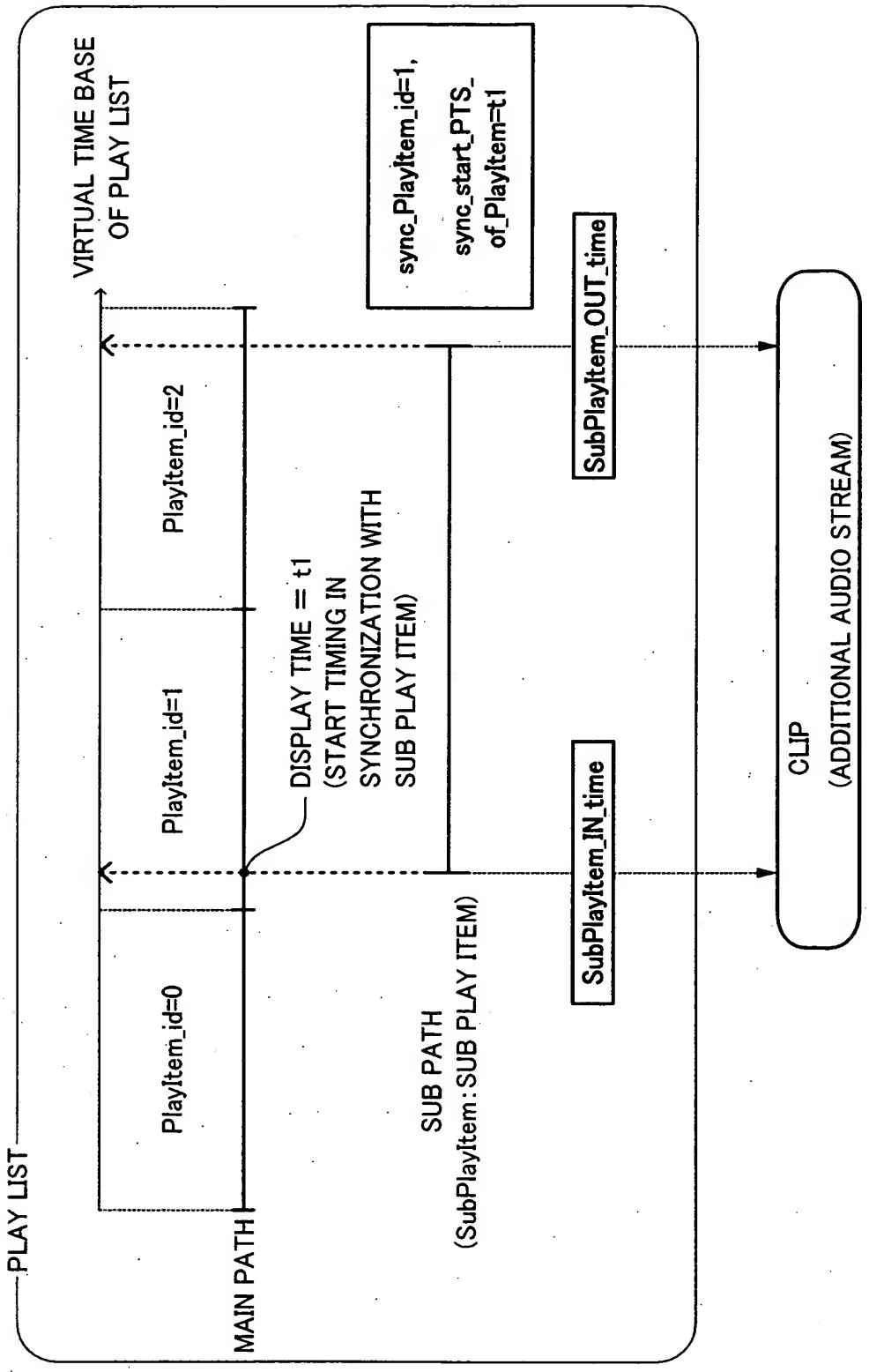

PLAY LIST
VIRTUAL TIME BASE OF PLAY LIST
PlayItem_id=0
PlayItem_id=1
PlayItem_id=2
MAIN PATH
DISPLAY TIME = t1
(START TIMING IN SYNCHRONIZATION WITH SUB PLAY ITEM)
SUB PATH
(SubPlayItem : SUB PLAY ITEM)
sync_PlayItem_id=1,
sync_start_PTS_of_PlayItem=t1
SubPlayItem_IN_time
SubPlayItem_OUT_time
CLIP
(ADDITIONAL AUDIO STREAM)

Fig. 48

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
zzzzz.clpi{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
SequenceInfo_start_address	32	uimsbf
ProgramInfo_start_address	32	uimsbf
CPI_start_address	32	uimsbf
ClipMark_start_address	32	uimsbf
reserved_for_future_use	128	bslbf
ClipInfo()		
for(i=0;i<N1;i++){		
padding_word	16	bslbf
}		
SequenceInfo()		
for(i=0;i<N2;i++){		
padding_word	16	bslbf
}		
ProgramInfo()		
for(i=0;i<N3;i++){		
padding_word	16	bslbf
}		
CPI()		
for(i=0;i<N4;i++){		
padding_word	16	bslbf
}		
ClipMark()		
for(i=0;i<N5;i++){		
padding_word	16	bslbf
}		
}		

Fig. 49

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
ClipInfo(){		
length	32	unimsbf
reserved	8	bslbf
application_type	8	unimsbf
Clip_stream_type	8	unimsbf
reserved	40	unimsbf
TS_recording_rate	32	unimsbf
num_of_source_packets	32	unimsbf
BD_system_use	1024	bslbf
TS_type_info_block()		
}		

Fig. 50

application_type	DESCRIPTION
0	CORRESPONDING m2ts FILE DOES NOT COMPLY WITH RULE OF HDMV TRANSPORT STREAM.
1	CORRESPONDING m2ts FILE COMPLIES WITH RULE OF HDMV TRANSPORT STREAM. (NORMAL HDMV STREAM)
2	CORRESPONDING m2ts FILE COMPLIES WITH RULE OF HDMV TRANSPORT STREAM FOR STILL PICTURE THAT SYNCHRONIZES WITH AUDIO REPRODUCTION. (TIME BASE SLIDE SHOW)
3	CORRESPONDING m2ts FILE COMPLIES WITH RULE OF HDMV TRANSPORT STREAM FOR STILL PICTURE THAT IS REPRODUCED NOT IN SYNCHRONIZATION WITH AUDIO. (BROWSABLE SLIDE SHOW)

Fig. 51

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
SequenceInfo(){		
length	32	unimsbf
reserved_for_word_align	8	bslbf
num_of_ATC_sequences	8	unimsbf
for(atc_id=0;atc_id<num_of_ATC_sequences;atc_id++){		
SPN_ATC_start *[atc_id]*	32	unimsbf
num_of_STC_sequences *[atc_id]*	8	unimsbf
offset_STC_id *[atc_id]*	8	unimsbf
for(*stc_id=offset_STC_id[atc_id];stc_id<(num_of_STC_sequences[atc_id]+offset_STC_id[atc_id]);stc_id++*){		
PCR_PID *[atc_id][stc_id]*	16	unimsbf
SPN_STC_start *[atc_id][stc_id]*	32	unimsbf
presentation_start_time *[atc_id][stc_id]*	32	unimsbf
presentation_end_time *[atc_id][stc_id]*	32	unimsbf
}		
}		
}		

Fig. 52

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
.ProgramInfo(){		
length	32	unimsbf
reserved_for_word_align	8	bslbf
num_of_program_sequences	8	unimsbf
for(i=0;i<*num_of_program_sequences*;i++){		
SPN_program_sequence_start *[i]*	32	unimsbf
program_map_PID *[i]*	16	bslbf
num_of_streams_in_ps *[i]*	8	unimsbf
num_of_groups *[i]*	8	unimsbf
for(*stream_index=0;stream_index<num_of_streams_in_ps[i];stream_index++*){		
stream_PID *[i][stream_index]*	16	unimsbf
StreamCodingInfo(*i,stream_index*)		
}		
}		
}		

Fig. 53

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
StreamCodingInfo(i,stream_index){		
length	8	bslbf
stream_coding_type	8	unimsbf
if(stream_coding_type==0x02){		
video_format	4	unimsbf
frame_rate	4	unimsbf
aspect_ratio	4	unimsbf
reserved_for_word_align	2	bslbf
cc_flag	1	unimsbf
reserved_for_word_align	1	bslbf
}else if(stream_coding_type==0x80//stream_coding_type==0x81//stream_coding_type==0x82){		
audio_presentation_type	4	unimsbf
sampling_frequency	4	unimsbf
language_code	16	bslbf
reserved_for_word_align	8	bslbf
}else if(stream_coding_type==0x90){		
language_code	16	bslbf
T.B.D		
}else if(stream_coding_type==0xA0){}		
language_code	16	bslbf
T.B.D		
}		
}		

Fig. 54

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
CPI(){		
length	32	unimsbf
reserved_for_word_align	12	bslbf
CPI_type	4	unimsbf
EP_map_for_BDMV()		
}		

Fig. 55

CPI_type	DESCRIPTION
0	RESERVED FOR FUTURE USE
1	EP_map TYPE
2	TU_map TYPE
3-7	RESERVED FOR FUTURE USE
8	EP_map TYPE FOR BDMV
9-15	RESERVED FOR FUTURE USE

Fig. 56

SYNTAX	DATA LENGTH (BITS)	MNEMONIC
EP_map_for_one_stream_PID(*EP_stream_type,Nc,Nf*){		
EP_fine_table_start_address	32	unimsbf
for(i=0;i<*Nc*;i++){		
ref_to_EP_fine_id*[i]*	18	unimsbf
PTS_EP_coarse*[i]*	14	unimsbf
SPN_EP_coarse*[i]*	32	unimsbf
}		
for(i=0;i<X;i++){		
padding_word	16	bslbf
}		
for(*EP_fine_id=0;EP_fine_id<Nf;EP_fine_id++*){		
is_angle_change_point*[EP_fine_id]*	1	bslbf
I_end_position_offset*[EP_fine_id]*	3	bslbf
PTS_EP_fine*[EP_fine_id]*	11	unimsbf
SPN_EP_fine*[EP_fine_id]*	17	unimsbf
}		
}		

Fig. 57

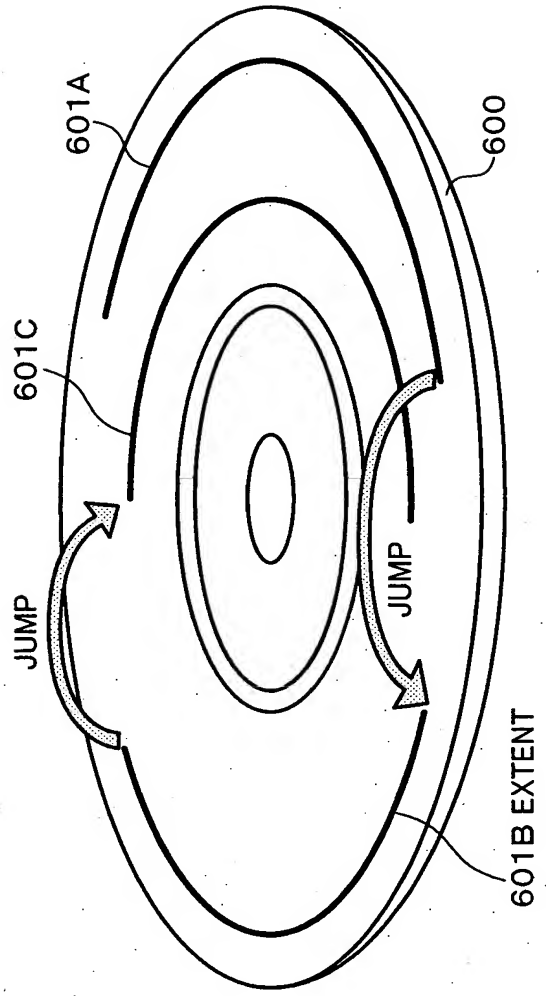
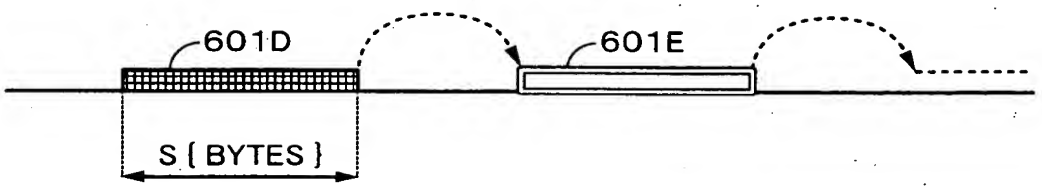


Fig. 58



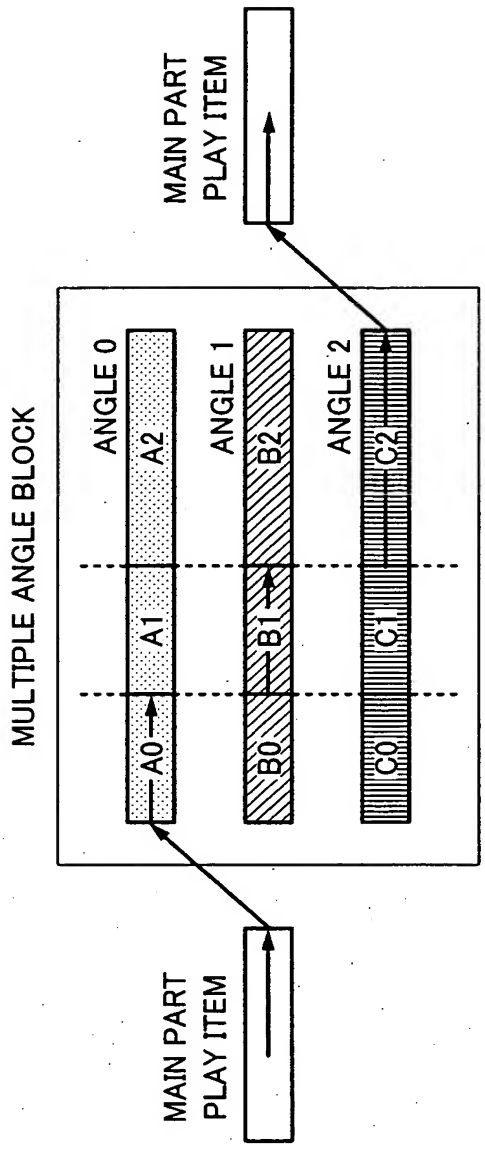


Fig. 59A

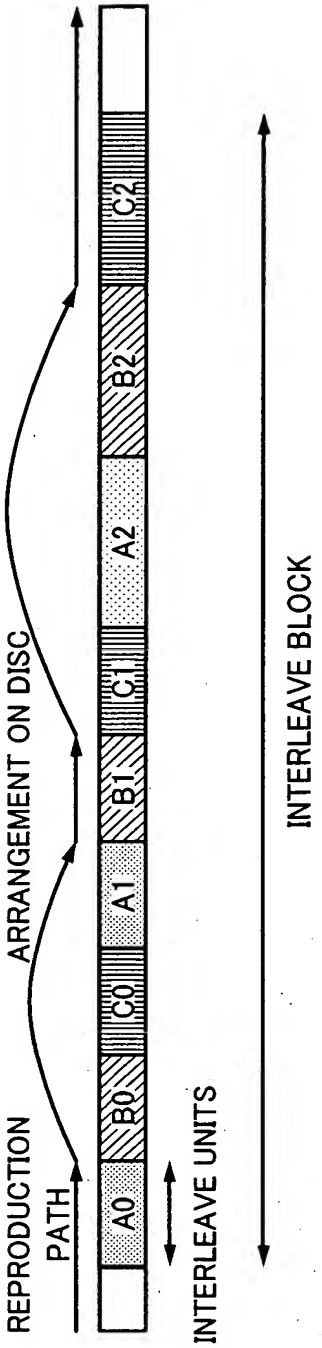


Fig. 59B

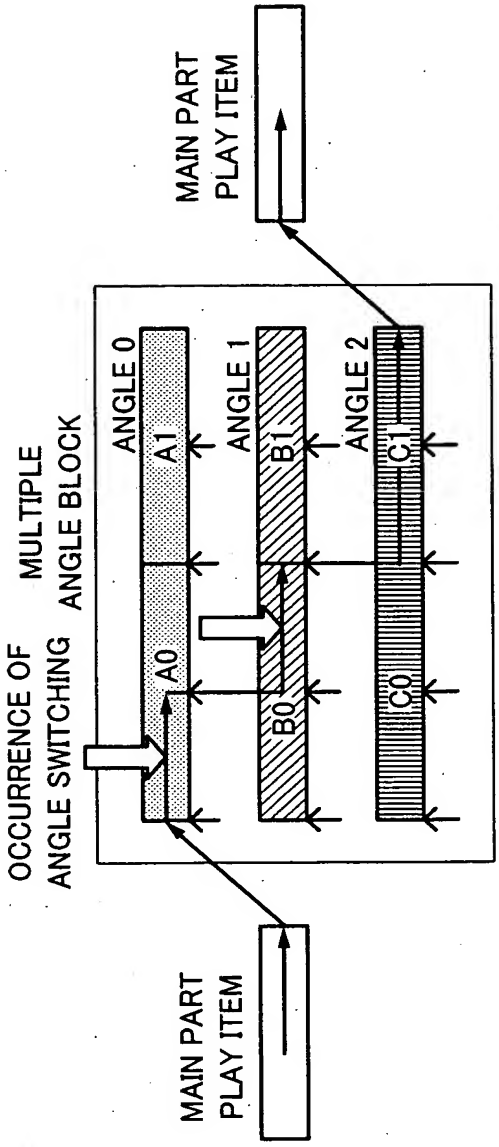


Fig. 60A

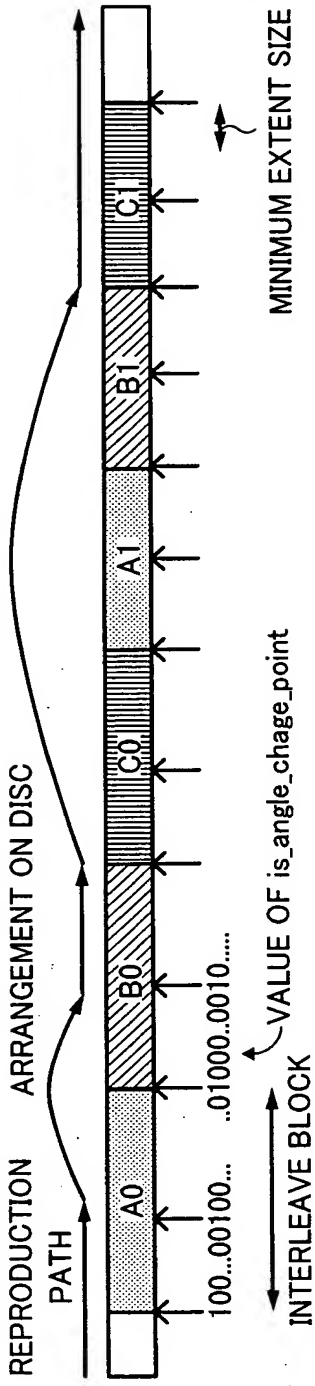


Fig. 60B

Fig. 61

is_angle_change_point	Meaning
0	THIS EP ENTRY DOES NOT CORRESPOND TO ANGLE SWITCHABLE POINT.
1	THIS EP ENTRY CORRESPONDS TO ANGLE SWITCHABLE POINT.

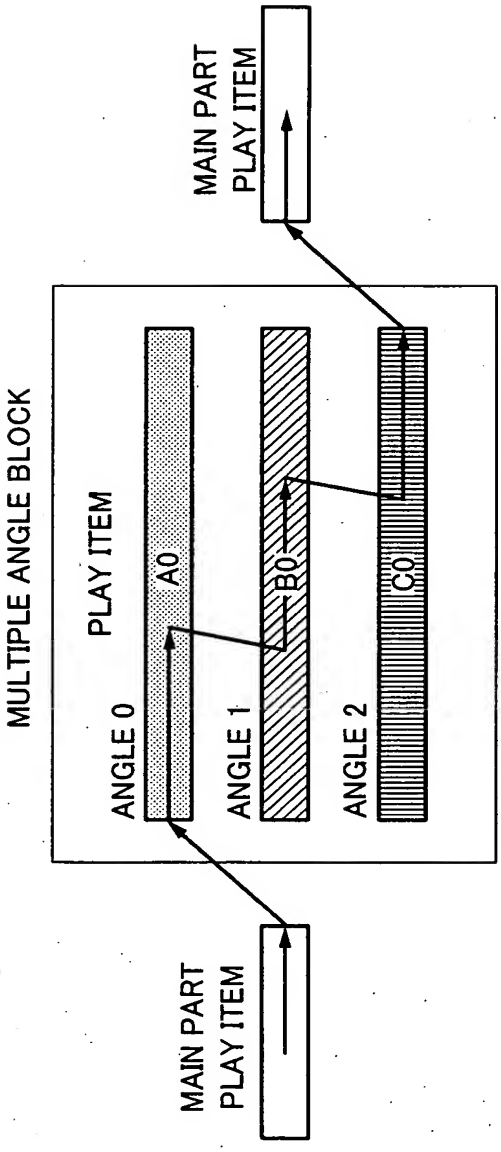


Fig. 62A

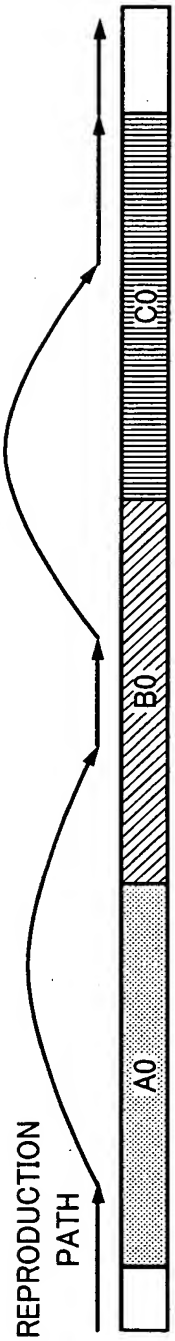


Fig. 62B

Fig. 63A

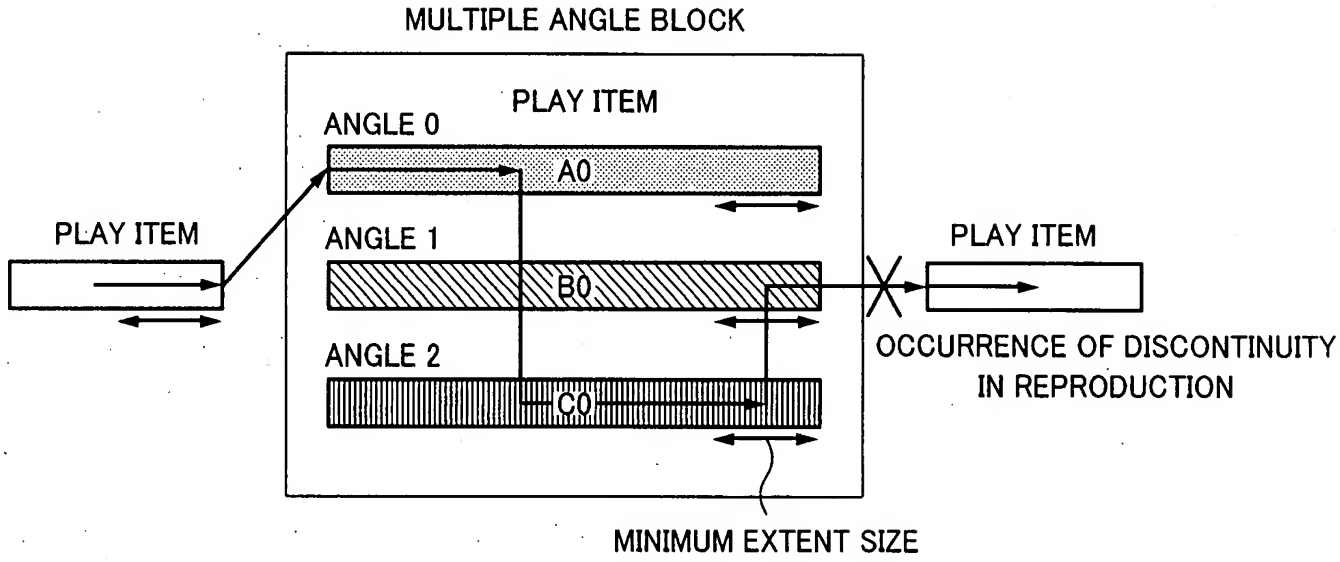


Fig. 63B

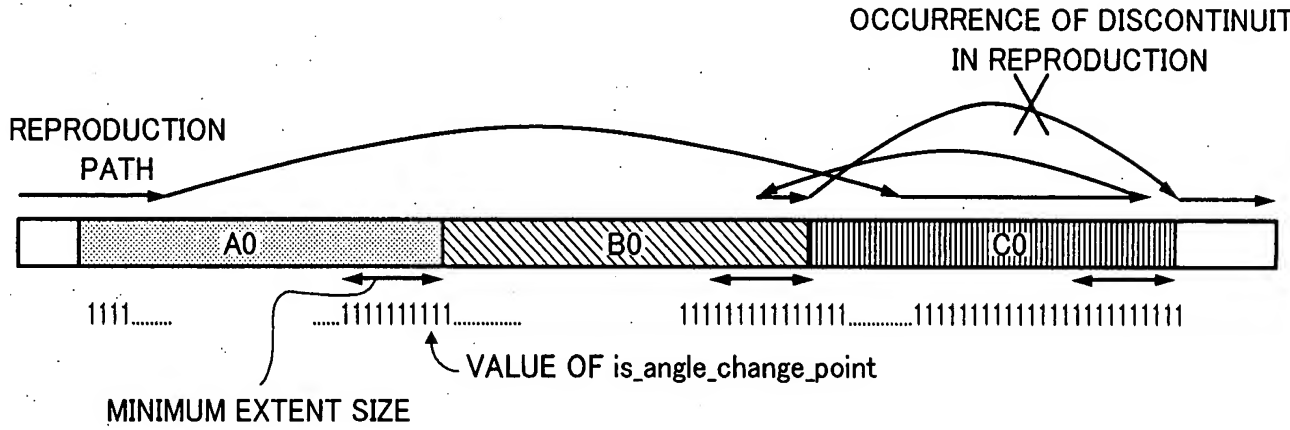


Fig. 64A

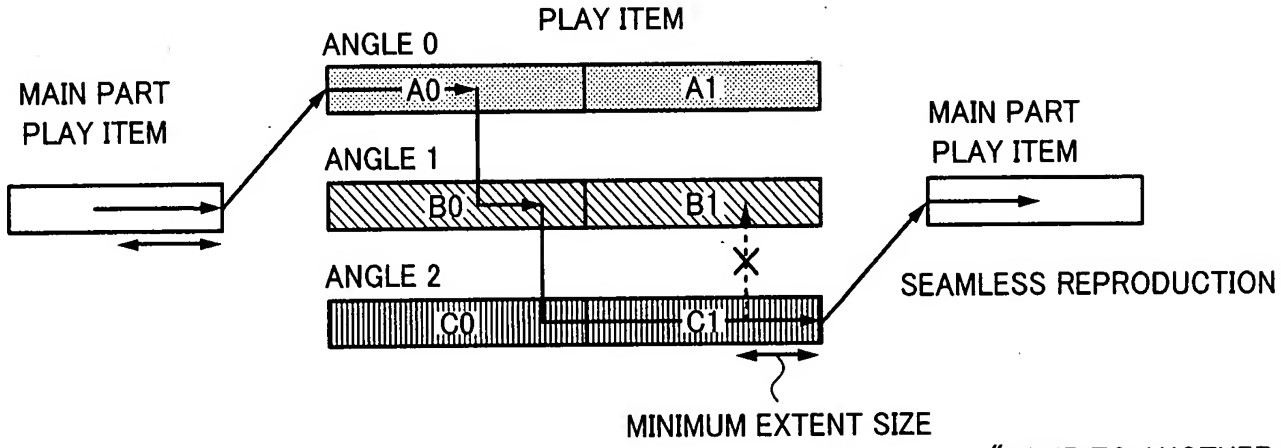


Fig. 64B

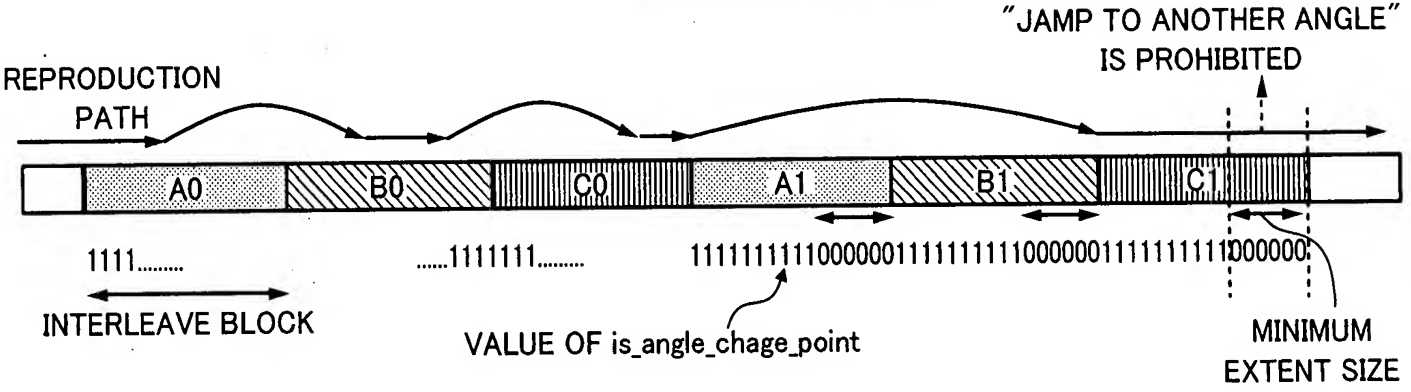


Fig. 65A

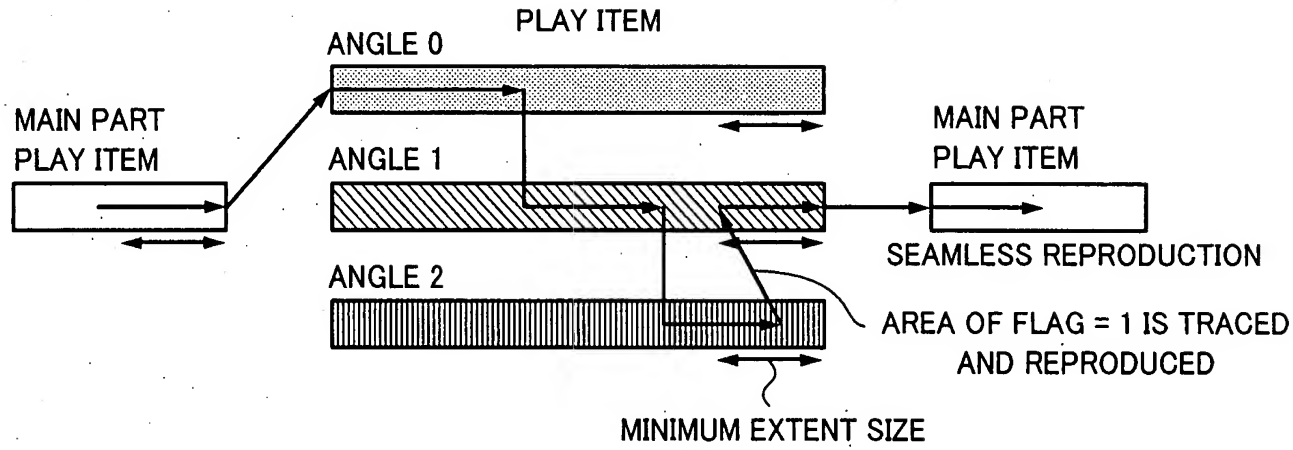


Fig. 65B

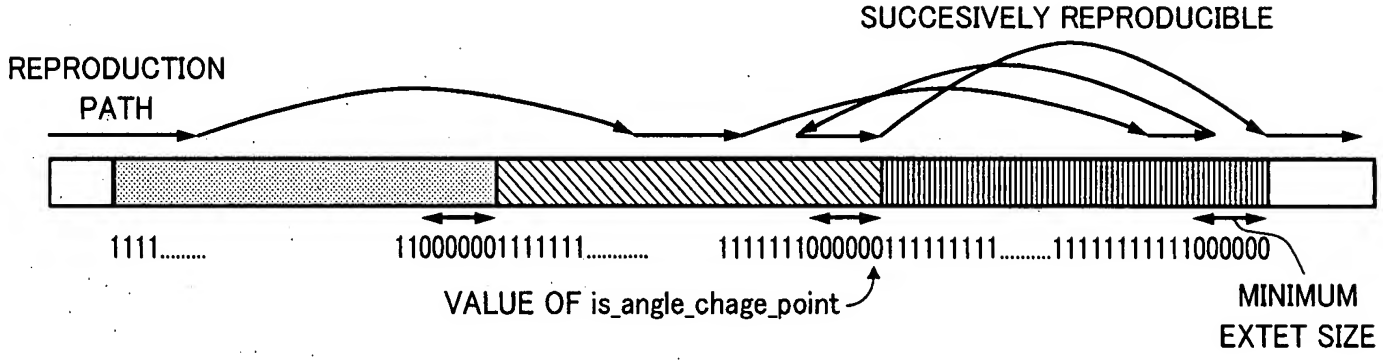


Fig. 66A

Fig. 66
Fig. 66A | Fig. 66B | Fig. 66C

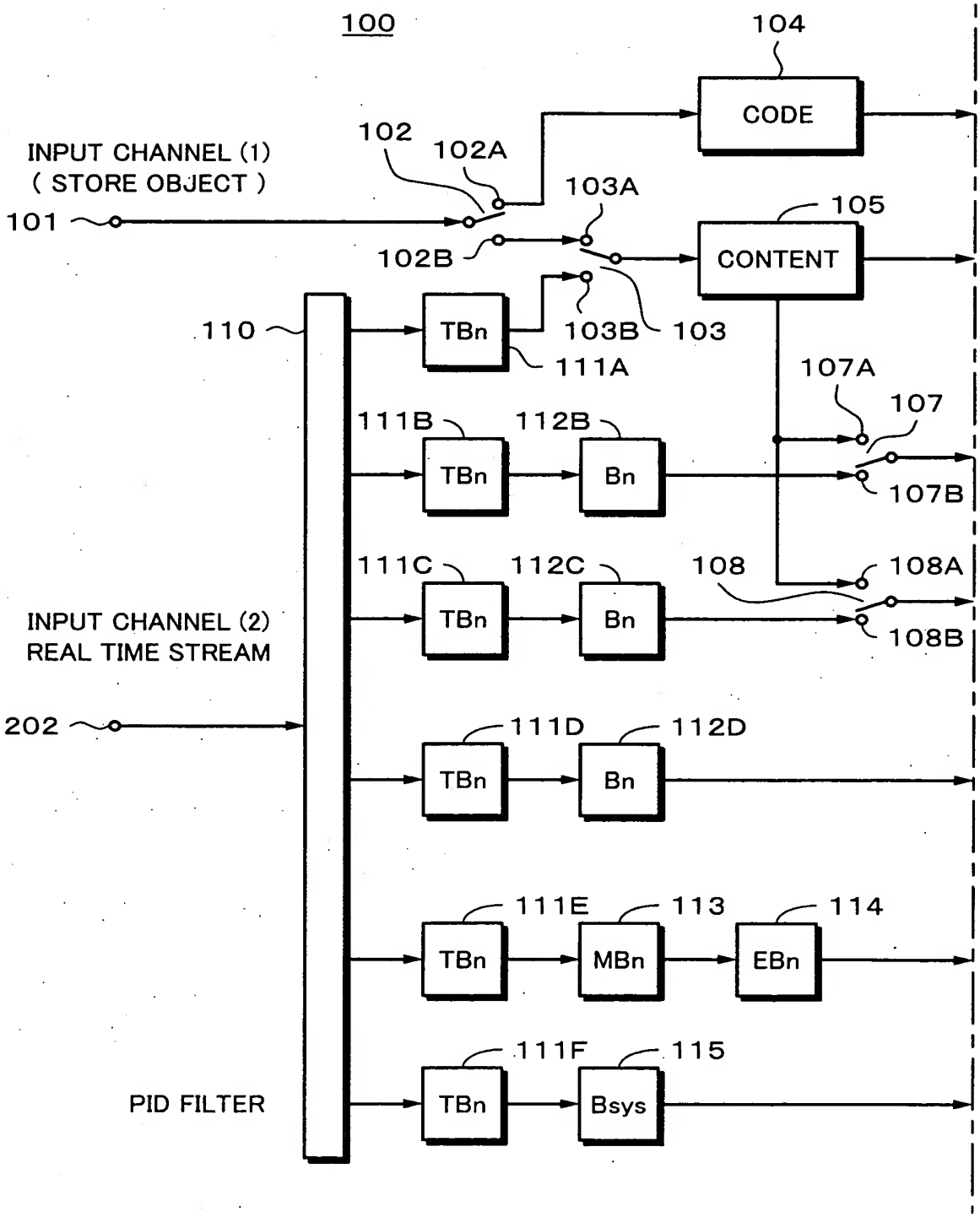


Fig. 66B

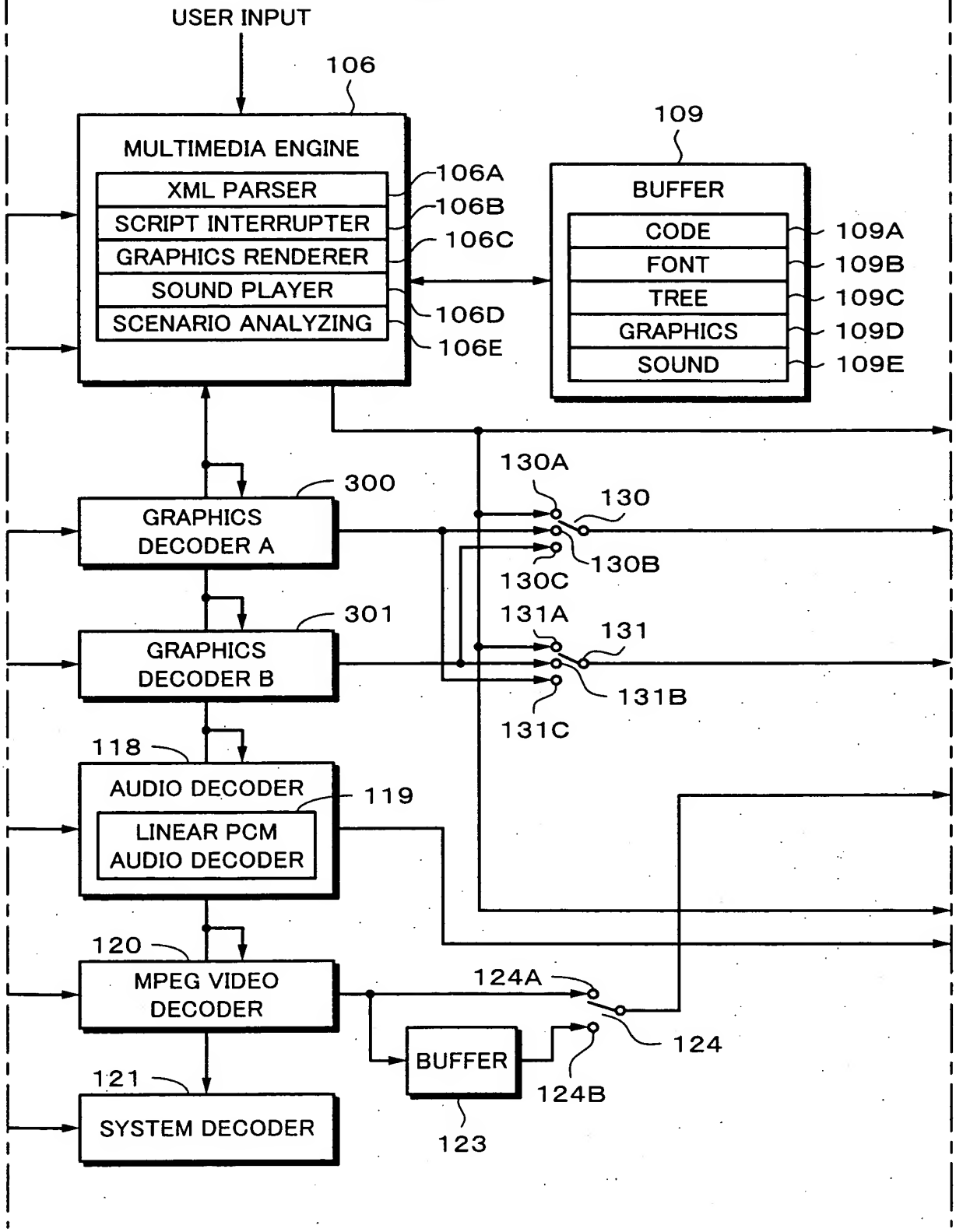
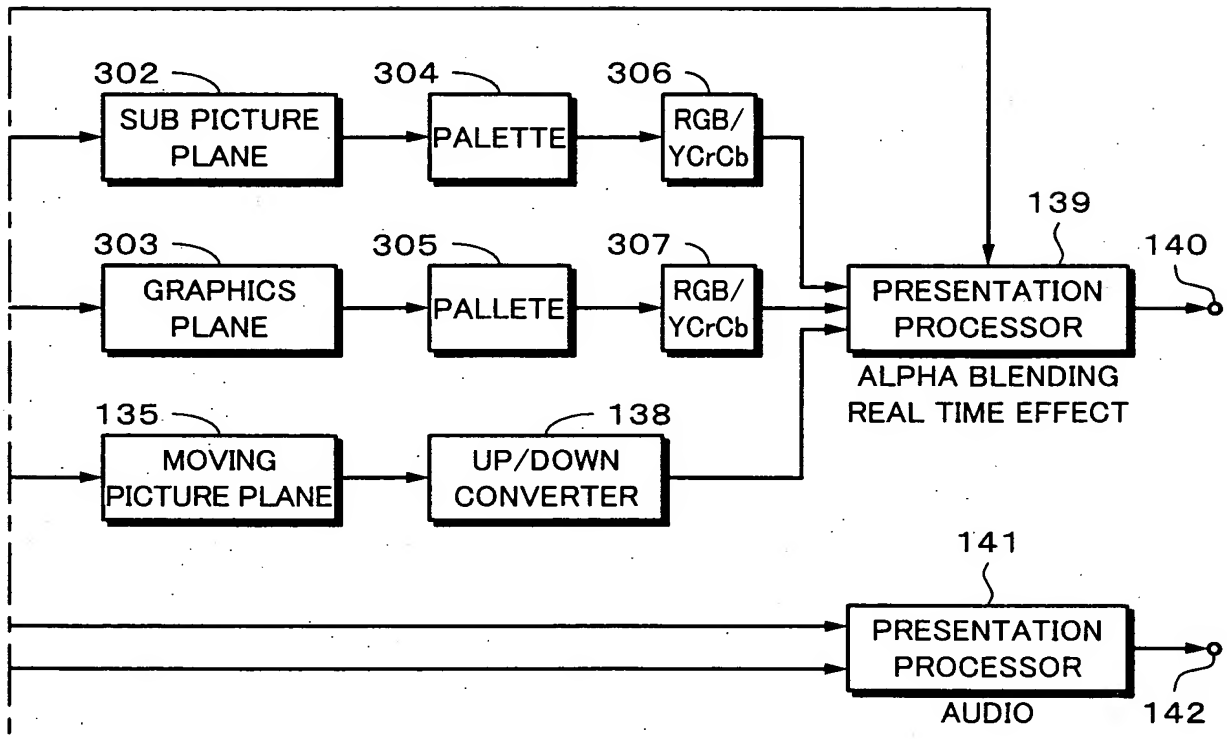

USER INPUT
106
MULTIMEDIA ENGINE
XML PARSER
SCRIPT INTERRUPTER
GRAPHICS RENDERER
SOUND PLAYER
SCENARIO ANALYZING
106A
106B
106C
106D
106E
109
BUFFER
CODE
FONT
TREE
GRAPHICS
SOUND
109A
109B
109C
109D
109E
300
GRAPHICS DECODER A
301
GRAPHICS DECODER B
118
AUDIO DECODER
LINEAR PCM AUDIO DECODER
119
120
MPEG VIDEO DECODER
121
SYSTEM DECODER
130A
130
130B
130C
131A
131
131B
131C
124A
124
BUFFER
124B
123

Fig. 66C


302
SUB PICTURE PLANE
304
PALETTE
306
RGB/ YCrCb
303
GRAPHICS PLANE
305
PALLETE
307
RGB/ YCrCb
139
140
PRESENTATION PROCESSOR
ALPHA BLENDING
REAL TIME EFFECT
135
MOVING PICTURE PLANE
138
UP/DOWN CONVERTER
141
PRESENTATION PROCESSOR
AUDIO
142

DESCRIPTION OF REFERENCE NUMERALS

10	MOVING PICTURE PLANE
11	SUBTITLE PLANE
12	GRAPHICS PLANE
22	PALETTE
30	BD VIRTUAL PLAYER
31	PLAYER COMMANDS
32	COMMON PARAMETER
40	PLAYBACK CONTROL PROGRAM
41	METHOD
60	MENU SCREEN
70	SCENARIO
73A - 73M	PLAY LIST
100	PLAYER DECODER
104	CODE BUFFER
105	CONTENTS BUFFER
106	MULTIMEDIA ENGINE
109	BUFFER
110	PID FILTER
116	SUB PICTURE DECODER
117	STILL PICTURE DECODER
118	AUDIO DECODER
120	MPEG VIDEO DECODER
132	SUB PICTURE PLANE
133	GRAPHICS PLANE

134	STILL PICTURE PLANE
135	MOVING PICTURE PLANE
226	PNG DECODER BUFFER
227	PNG DECODER
228	OBJECT BUFFER
229	PLANE BUFFER
231	AUDIO MIXER
500	GRAPHICS DECODER A
501	GRAPHICS DECODER B
502	SUB PICTURE PLANE
503	GRAPHICS PLANE
601A, 601B, 601C, 601D, 601E	EXTENT